PolicyVis: Firewall Security Policy Visualization and Inspection

Tung Tran, Ehab Al-Shaer, and Raouf Boutaba – University of Waterloo, Canada

ABSTRACT

Firewalls have an important role in network security. However, managing firewall policies is an extremely complex task because the large number of interacting rules in single or distributed firewalls significantly increases the possibility of policy misconfiguration and network vulnerabilities. Moreover, due to low-level representation of firewall rules, the semantic of firewall policies become very incomprehensible, which makes inspecting of firewall policy's properties a difficult and error-prone task.

In this paper, we propose a tool called PolicyVis which visualizes firewall rules and policies in such a way that efficiently enhances the understanding and inspecting firewall policies. Unlike previous works that attempt to validate or inspect firewall rules based on specific queries or errors, our approach is to visualize firewall policies to enable the user to place general inquiry such as "does my policy do what I intend to do" unrestrictedly. We describe the design principals in PolicyVis and provide concepts and examples dealing with firewall policy's properties, rule anomalies and distributed firewalls. As a result, PolicyVis considerably simplifies the management of firewall policies and hence effectively improves the network security.

Introduction

Network security is essential to the development of internet and has attracted much attention in research and industrial communities. With the increase of network attack threats, firewalls are considered effective network barriers and have become important elements not only in enterprise networks but also in small-size and home networks. A firewall is a program or a hardware device to protect a network or a computer system by filtering out unwanted network traffic. The filtering decision is based on a set of ordered filtering rules written based on predefined security policy requirements.

Firewalls can be deployed to secure one network from another. However, firewalls can be significantly ineffective in protecting networks if policies are not managed correctly and efficiently. It is very crucial to have policy management techniques and tools that users can use to examine, refine and verify the correctness of written firewall filtering rules in order to increase the effectiveness of firewall security.

It is true that humans are well adapted to capture data essences and patterns when presented in a way that is visually appealing. This truth promotes visualization on data, on which the analysis is very hard or ineffective to carry out because of its huge volume and complexity. The amount of data that can be processed and analyzed has never been greater, and continues to grow rapidly. As the number of filtering rules increases largely and the policy becomes much more complex, firewall policy visualization is an indispensable solution to policy management. Firewall policy visualization helps users understand their policies easily and grasp complicated rule patterns and behaviors efficiently.

In this paper, we present PolicyVis, a useful tool in visualizing firewall policies. We describe design principals, implementations and application examples that deal with discovering firewall policy's properties, rule anomalies for single or distributed firewalls.

Although network security visualization has been given strong attention in the research community, the emphasis was mostly on the network traffic [4, 8]. On the other hand, tools in [12, 16] visualize some firewall aspects, but don't give users a thorough look at firewall policies.

This paper is organized as follows. In the next section, we summarize related work. We then describe PolicyVis design principals followed by descriptions of scenarios that show the usefulness of PolicyVis. Next, we show how rule anomalies are visualized by PolicyVis and demonstrate some examples of determining rule anomalies by using PolicyVis. We then describe visualization on distributed firewalls in PolicyVis followed by a disucssion of the implementation and evaluation of PolicyVis. Finally, we show conclusions and plans for future work.

Related Work

A significant amount of work has been reported in the area of firewall and policy-based security management. In this section, we focus our study on the work that closely related to PolicyVis' design objectives: network security visualization and policy management.

There are many visualization tools introduced to enhance network security. PortVis [15] uses port-based detection of security activities and visualizes network

traffic by choosing important features for axes and displaying network activities on the graph. SeeNet [3] supports three static network displays: two of these use geographical relationships, while the third is a matrix arrangement that gives equal emphasis to all network links. NVisionIP [11] uses a graphical representation of a class-B network to allow users to quickly visualize the current state of networks. Le Malécot, et al. [14] introduced an original visualization design which combines 2-D and 3-D visualization for network traffic monitoring. However, these tools only focus on visualizing network traffic to assist users in understanding network events and taking according actions.

Moreover, previous work on firewall visualization only concentrates on visualizing how firewalls react to network traffic based on network events. Chris P. Lee, et al. [12] proposed a tool visualizing firewall reactions to network traffic to aid users in configuration of firewalls. FireViz [16] visually displays activities of a personal firewall in real time to possibly find any potential loop holes in the firewall's security policies. These tools can only detect a small subset of all firewall behaviors and can not determine all possible potential firewall patterns by looking at the policy directly like PolicyVis. Besides, Tufin SecureTrack [19] is a commercial firewall operations management solution, however, it provides change management and version control for firewall policy update. It basically visualizes firewall policy version changes, but not rule properties and relations and allows users to receive alerts if any change violates the organizational policy. Thus, Tufin SecureTrack can not be used for rules analysis and anomaly discovery.

In the field of firewall policy management, a filtering policy translation tool proposed in [2] describes, in a natural textual language, the meaning and the interactions of all filtering rules in the policy, revealing the complete semantics of the policy in a very concise fashion. However, this tool is not as efficient as PolicyVis in helping users capture the policy properties quickly in case of huge number of rules in the policy. In [1], the authors mentioned firewall policy anomalies and techniques to discover them, and suggested a tool called Firewall Policy Advisor which implements anomaly discovery algorithms. However, Firewall Policy Advisor is not capable of showing all potential behaviors of firewall policies and doesn't help users in telling if a policy does what he wants.

The authors in [6, 10] suggest methods for detecting and resolving conflicts among general packet filters. However, only correlation anomaly [1] is considered because it causes ambiguity in packet classifiers. In addition, the authors in [13, 18] proposed firewall analysis tools that allow users to issue customized queries on a set of filtering rules and display corresponding outputs in the policy. However, the query reply could be overwhelming and still complex to understand. PolicyVis output is more comprehensible. Moreover, those tools require users to consider very specific issues to inspect the policy. PolicyVis, on the other hand, enables users to investigate the policy at once, which is more practical and efficient in large policies.

PolicyVis Objectives

Information visualization is always an effective way to capture and explore large amount of abstract data. With the necessity of guaranteeing a correct firewall behavior, users need to recognize and fix firewall misconfigurations in a swift manner. However, the complexity of dealing with firewall policies is they are attributed to the large number of rules, rules complexity and rules dependency. Those facts motivate a tool which visualizes all firewall rules in such a way that rule interactions are easily grasped and analyzed in order to come up with an opportune solution to any firewall security breach.

PolicyVis is a visualization tool of firewall policies helps users to achieve the following goals in an effective and fast fashion:

- **Visualizing rule conditions, address space and action**: a firewall policy is attributed by rules format, rules dependency and matching semantics. Comprehensive visualization of firewall policies requires a mechanism of transforming firewall rules to visual elements which significantly enhance the investigation of policies. PolicyVis effectively visualizes all firewall rule core elements: conditions, address space and action.
- **Firewall policy semantic discovery**: it is a very normal demand of users to know all possible behaviors of a firewall to its intended protected system. With advantages of visualization and many graphic options supported by PolicyVis, all potential firewall behaviors can be easily discovered, which are normally very hard to grasp in a usual context.
- **Firewall policy rule conflict discovery**: PolicyVis can be able to not only give users a view on normal rule interactions, but also pinpoint all possible rule anomalies in the policy. This is a crucial feature of PolicyVis to become a very helpful tool for users. All kind of rule conflicts can be efficiently visualized without worrying about running any algorithm to find potential rule conflicts.
- **Firewall policy inspection based on users' intention**: with a policy of thousands of rules, it is much likely that the user will make configuration mistakes (not rule conflicts mentioned above) in the policy which causes the firewall to function incorrectly. PolicyVis brings all firewall rules to a graphic view so that all configuration mistakes are highlighted without any difficulty.
- **Visualizing distributed firewalls**: distributed firewalls security is as important as a single firewall, besides visualizing a single firewall.

PolicyVis also lets users visualize distributed firewalls with the same efficiency in all goals mentioned above.

PolicyVis Design Principles

The fundamental design requirements for PolicyVis included:

- **Simplicity**: It should be fairly intuitive for users to inspect firewall policies in a 2D graph using multiple fields. We chose to compress firewall rules into 2D graph with three factors because it is much likely that a certain field (like source port) can be ignored or not important when investigating the policy. 2D graph is simple but quite effective in terms of helping users thoroughly look into the policy's behaviors.
- **Expressiveness**: It is very important that users can easily capture true rule interactions so that appropriate actions can be taken immediately. PolicyVis supports very detailed and thorough visualization of all possible firewall rules' behaviors by considering all rules fields, rule orders as well as all rule actions.
- **Flexible Visualization scope**: PolicyVis allows users to visualize what they are interested in (the target: any rule field) so that all possible aspects of the policy can be viewed and analyzed. Moreover, with multiple dimensions support, PolicyVis is flexible in letting users choose desired fields for the graph coordinates, which is convenient and effective to observe and investigate the policy from different views. Besides, there are choices on type of traffic (accepted, denied or both) which can be viewed separately to meet users' different purposes.
- **Ability to Compress, Focus and Zoom**: It is a normal thing to take a closer look at a specific set of rules when investigating the policy. PolicyVis supports zooming so that users can closely investigate a set of considered rules. This zooming feature is very useful if too many rules get involved in the investigation and the axes get crowded. In addition, PolicyVis gives users the ability to investigate rule anomalies existing in the policy through the focusing feature. With PolicyVis, users can also visualize the whole policy at once as well as portions of the policy partitioned by ranges of a specific field. This is a helpful feature of PolicyVis when users want to consider the policy applied to a subnet or a desired portion of the network.
- **Ability to use policy segmentation**: In order to investigate accepted or denied traffic only, policy segmentation with BDDs technique [5] is a powerful means employed by PolicyVis to increase the effectiveness and correctness of extracting useful information from the policy.
- **Ability to use symbols, colors, notations**: Policies are attributed by rules format, rules dependency and matching semantics (rule order). Moreover, firewall rules contain conditions (protocol, port and address), values (specific and wildcard) and actions (allowed and denied). PolicyVis visualizes those features using colors, symbols, and notations which are essential for users to capture quickly and easily the inside interactions and performance of firewall policies.

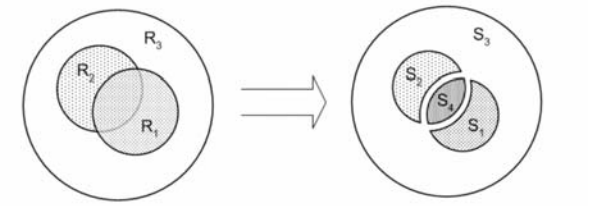
Multi-level Visualizing of Firewall Policies

Using PolicyVis, multi-level visualizing of firewall policies can be accomplished effectively. With PolicyVis' many flexible features, user can inspect the firewall policy from different views (like port level, address level, etc.) to understand all potential inside behaviors of the policy. In order to achieve this goal, PolicyVis deploys many methods and techniques which efficiently bring firewall policies to expressive visual views.

Using BDDs To Segment Policy and Find Accepted and Denied Spaces

Firewall policy segmentation using Binary Decision Diagram or BDD was first introduced by our group in [5, 9] to enhance the firewall validation and testing procedures. As defined in [5], a segment is a subset of the total traffic address space that belongs to one or more rules, such that each of the member elements (i.e., header tuples) conforms to exactly the same set of policy rules. Rules and address spaces are represented as Boolean expressions and BDD is used as a standard canonical method to represent Boolean expressions. By taking advantages of BDD's properties, firewall rules are effectively segmented into disjoint segments each of which belong to either accepted or denied space.

R1: tcp	121.63.*.*: any	143.91.78.*: any	accept
R2: tcp	121.63.71.*: any	143.91.*.*: any	accept
R3: any	*.*.*.*: any	*.*.*.*: any	deny



(a) Policy address-space (b) Segmented address-space

Table 1: Example of firewall policy segmentation.

In specific, the authors in [9] suggest constructing a Boolean expression for a policy P_a using the rule constraints as follows:

$$P_a = \bigvee_{i \in index(a)} (\neg C_1 \wedge \neg C_2 \wedge \cdots \wedge \neg C_{i-1} \wedge \neg C_i)$$

where *index(a)* is the set of indices of rules that have *a* as their action and C_i is the rule condition of conjunctive fields. In other words,

$$index(a) = \{\ i \mid R_i = C_i \rightsquigarrow a\ \}$$

This formula can be understood as saying that a packet will trigger action a if it satisfies the condition Ci for some rule Ri with action a, provided that the packet does not match the condition of any prior rule in the policy. Table 1 shows an example of a policy of three intersecting rules forming total of four independent segments of policy address space.

PolicyVis allows users to visualize only accepted or denied traffic; therefore it is important to efficiently extract those spaces from the policy. A naïve algorithm to achieve this might take exponential running time. Fortunately, policy segmentation using BDD is quite effective in doing this. We decided to employ BDDs for segmenting rules to quickly retrieve correct accepted and denied spaces. This makes PolicyVis a reliable and fast tool. Policy rules are segmented using BDD right after they are read from the input file. This ahead-of-time rule segmentation speed up the process when the user chooses to visualize only accepted or denied traffic.

Firewall Visualization Techniques

In this section, we describe visualization techniques and methods used in our PolicyVis tool to achieve the objectives. More specific techniques and algorithms to visualize firewall anomalies are described later.

To achieve the visualization effectiveness, PolicyVis supports both policy segments and policy rules visualization, which depends on properties of the policy users want to examine. When dealing with only accepted or denied space, PolicyVis visualizes policy segments obtained from using BDD as mentioned earlier. However, when users choose to investigate both accepted and denied spaces together, PolicyVis visualizes policy rules because the union of both spaces returns to the original rules. Moreover, visualizing policy rules in this case helps users capture all possible rule interactions which is hard to conceive by looking at separate visualizations of both spaces.

When users investigate a firewall policy scope (a field and a value), PolicyVis collects all rules (or segments) that have the corresponding field as a superset of the scope input and visualizes those rules (or segments). When choosing a scope to investigate, users want to inspect how the firewall policy applies to that scope, thus rules (or segments) that include only the address space of the target scope. Rules (or segments) are represented as rectangles with different colors to illustrate different kinds of traffic (accepted or denied). Those colors are set transparent so that rules overlapping with the same or different actions can be effectively recognized. Moreover, different symbols (small square and circle) placed at the corner of rectangles are used for different traffic protocols (e.g., TCP, UDP, ICMP, IGMP) and notations (i.e., tooltips or legends) are used to determine rules' order and related things.

When multiple rectangles (rules or segments) are sketched from the same coordinates, colors and symbols might not be enough to tell what kind of traffic or

Order	Prot	SrcIP	SrcPort	DestIP	DestPort	Action
1	tcp	140.192.37.2	*	161.120.33.*	1433	ACCEPT
2	tcp	140.192.37.1	*	161.120.33.41	22	DENY
3	tcp	140.192.37.*	*	161.120.33.41	22	ACCEPT
4	tcp	140.192.37.4	*	161.120.33.44	1433	ACCEPT
5	tcp	140.192.37.5	*	161.120.33.44	1433	ACCEPT
6	tcp	140.192.37.*	*	161.120.33.44	1433	DENY
7	tcp	140.192.38.3	*	161.120.33.44	22	DENY
8	tcp	140.192.38.8	*	161.120.33.44	22	DENY
9	tcp	140.192.38.*	*	161.120.33.44	22	ACCEPT
10	tcp	140.192.36.2	*	161.120.34.45	22	DENY
11	tcp	140.192.36.*	*	161.120.34.45	22	ACCEPT
12	tcp	141.192.36.*	*	161.121.33.*	23	DENY
13	tcp	141.192.*.*	*	161.121.33.*	23	ACCEPT
14	tcp	141.192.37.3	*	161.121.34.3	80	DENY
15	tcp	141.192.37.5	*	161.121.34.3	80	DENY
16	tcp	141.192.37.*	*	161.121.34.3	80	ACCEPT
17	tcp	141.193.38.*	*	161.121.34.3	21	ACCEPT
18	tcp	141.193.39.*	*	161.121.34.3	21	ACCEPT
19	tcp	141.192.*.*	*	161.121.34.4	21	ACCEPT
20	tcp	141.*.*.*	*	161.121.34.5	25	ACCEPT
21	udp	142.192.*.*	*	161.122.33.43	69	ACCEPT
22	udp	143.192.*.*	*	161.122.33.43	69	DENY
23	udp	144.*.*.*	*	161.122.33.43	69	ACCEPT
24	udp	145.*.*.*	*	161.122.33.43	69	ACCEPT

Figure 1: A single firewall policy.

protocol a rectangle belongs to. Additional notations are used to clearly indicate those properties. Round brackets are used to tell if a rectangle represents denied traffic, otherwise it represents allowed traffic. Curly brackets are used to denote UDP protocol, otherwise it is TCP protocol.

PolicyVis uses three different rule fields to build the policy graph, two of which are used as the graph's vertical and horizontal coordinates and the third field is integrated into the visualization objects (e.g., at the corner of rectangles) avoiding 3D graphs for simplicity. In general, by default, PolicyVis chooses the investigated scope as one of the coordinates (axes), and from three remaining fields, the least common field will be the other coordinate and the second least common field will be the last dimension.

Besides, PolicyVis places rule field values along x-axis and y-axis in such a way that the aggregated values (e.g., wildcards) precedes the discrete values in the axis, or closer to the origin of the graph. Moreover, the width, the length and the position of a rectangle are chosen based on its corresponding rule's attributes so that an aggregated rule or segment (represented by a rectangle) contains its subset ones in the graph and disjoint segments or rules are represented by non-overlapping rectangles (there are no adjacent rectangles).

Case Studies

In this section, we created application scenarios to explore the potential of PolicyVis to help users find the policy misconfiguration problems. All the scenarios were created based on the single firewall policy shown in Figure 1.

Scenario 1: The admin receives an email from the SSH server development team mentioning that there currently exists a SSH server zero-day exploit in the wild. He wants to investigate the firewall policy for accepted traffic to port *22*. The admin performs this investigation by choosing the target (scope): *Destination Port* with *22* as the input and viewing allowed traffic only as shown in Figure 2.

Observation: policy segments that allow traffic to SSH (port *22*) are extracted and visualized by PolicyVis as shown in Figure 2. Thus, the admin can then decide to block this traffic temporarily.

Scenario 2: The University's student database is stolen and the database server with IP address *161.120.33.44* (possibly compromised) is suspected not protected well by the firewall. The admin wants to investigate the firewall policy applied to this server. He looks into the traffic allowed and blocked by the firewall for this IP address by choosing the target (scope): *Destination Address* with *161.120.33.44* as the input as shown in Figure 3.

Observation: denied and allowed traffic to port *1433* (MSSQL server) controlled by the firewall is almost like what the admin expected except the traffic from source address *140.192.37.2* (from rule number **1**) which should not be allowed. The problem is traffic allowed to *161.120.33.** from rule **1** is also allowed to *161.120.33.44*. Thus, the admin might remove or change Rule **1** from the firewall.

Scenario 3: The University's whole network is down because of a denial of service attack. The admin suspects that this attack is from a specific region in a

Figure 2: Allowed traffic to port 22.

country with network IP address starting with *141.*.*.* aiming at many services including telnet, web, ftp, etc. He needs to revise the firewall policy for any traffic from any IP address starting with *141.*.*.*. The admin chooses the target (scope): *Source Address* with *141.*.*.** as the input and selects *Destination Port* (corresponding to University's network services) as one of the graph dimensions as shown in Figure 4.

Observation: the firewall policy currently blocks traffic to telnet service (port *23*) and web service (port *80*) from some IP addresses starting with *141*, however, SMTP service (port *25*) and FTP service (port *21*) are accessible from most of IP addresses starting with *141* and hence vulnerable to the attack. Thus, the admin may set firewall rules to block traffic from some or all addresses starting with *141* to those services as well.

Scenario 4: The University maintains two replicated TFTP servers (port *69*) with IP addresses *161.122.33.43* and *161.122.33.44* to satisfy students' high demand of downloading video lectures and also increase

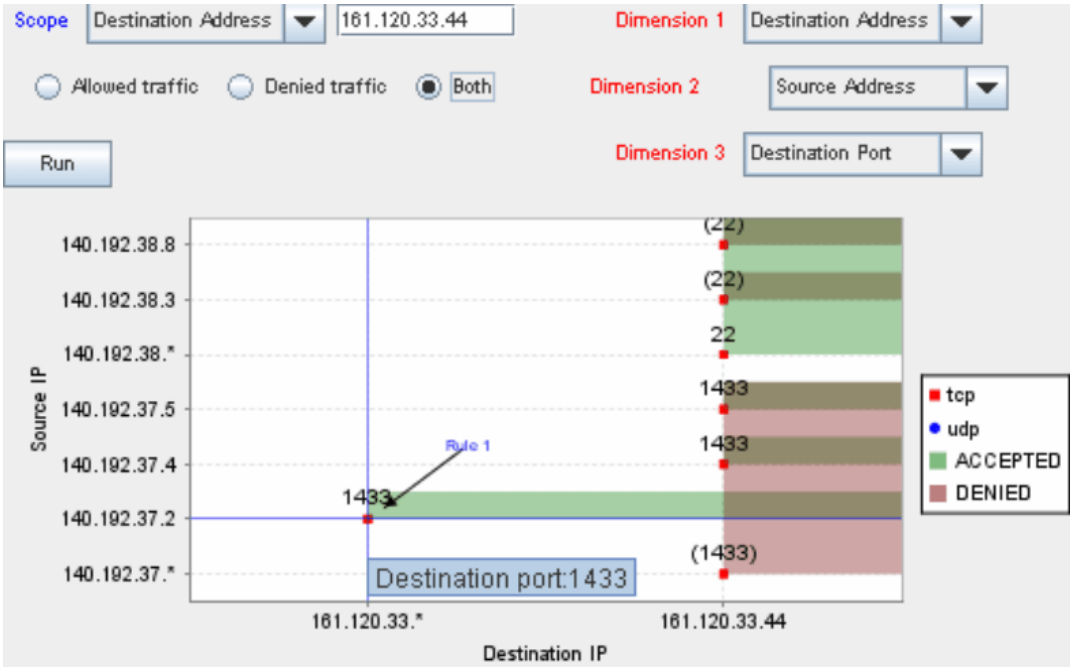


Figure 3: Traffic blocked and allowed to 161.120.33.44.

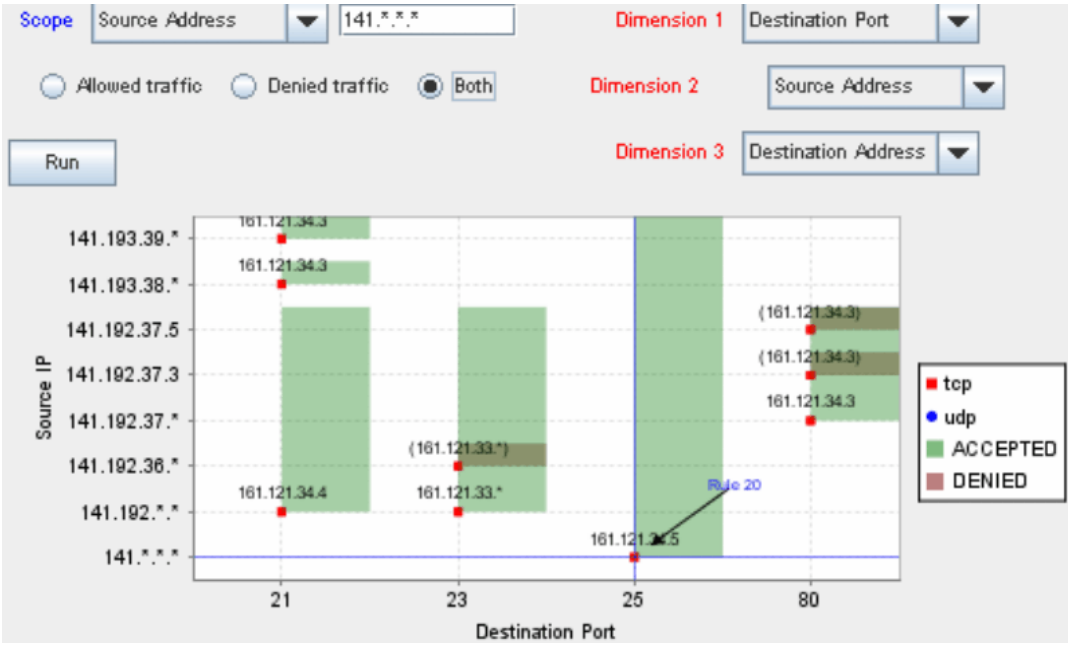


Figure 4: Controlled traffic from 141.*.*.*.

the downloading speed. However, several students still complain about low downloading speed and sometimes they are blocked from downloading. The admin first checks the two servers and sees that they both are working well. He suspects that he might make mistakes when writing firewall rules for the two servers so that one of them might not function as wanted. He needs to check the firewall policy and expects that the policy for both servers should be the same because they are replicated and have the same mission. The admin chooses the target (scope): *Destination Port* with *69* as the input as shown in Figure 5.

Observation: traffic controlled by the firewall to the two servers is not the same. The admin recognizes that he made mistakes blocking traffic from *144.*.*.** and *145.*.*.** to server *161.122.33.44* when they should be allowed as to server *161.122.33.43*. Thus, the admin corrects his mistakes by changing the actions in the corresponding rules in the firewall.

Visualizing Rule Anomalies

Definition

In this section, we mention crucial definitions and concepts of firewall policy anomalies introduced in [1] so that readers can understand how PolicyVis visualizes rule anomalies described in the next section.

A firewall policy conflict is defined as the existence of two or more filtering rules that may match the same packet or the existence of a rule that can never match any packet on the network paths that cross the firewall, e.g.:

- **Shadowing anomaly**: a rule is shadowed when a previous rule matches all the packets that match this rule and the shadowed rule will never be activated.
- **Generalization anomaly**: a rule is a generalization of a preceding rule if they have different actions and if the second rule can match all the packets that match the first rule.
- **Redundancy anomaly**: a redundant rule performs the same action on the same packets as another rule such that if the redundant rule is removed, the security policy will not be affected.
- **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.

Rule Anomaly Visualization Methodology and Algorithm

As the number of firewall rules increases, it is very likely that an anomaly will exist in the policy which threatens the firewall's security. Anomaly discovery is necessary in order to ensure the firewall's concreteness. Firewall policy advisor [1] is the first tool to discover anomalies in a firewall policy. However, it is not as expressive as PolicyVis in anomaly discovery and doesn't give users a visual view on how an anomaly occurs.

Four classes of firewall policy anomalies mentioned previously are visualized by PolicyVis. These anomalies are easily pinpointed by overlapping areas on the graph because an overlapping area represents for rules with overlapping traffic, which can potentially cause firewall policy anomalies. Each of the anomalies has specific features that are easily recognized on the

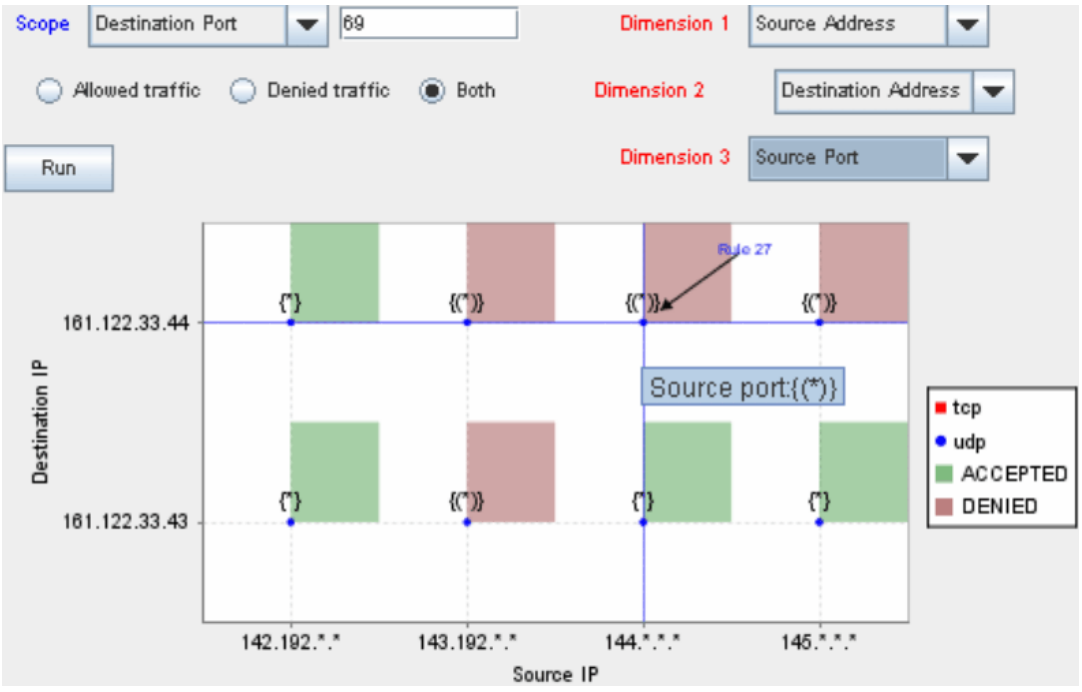


Figure 5: Firewall policy for destination port 69.

PolicyVis graph because its corresponding overlapping area is formed (or look) differently in terms of rectangles position, colors and notations. These features are different for all four anomalies.

As PolicyVis visualizes rules in 2D-graph which shows users only three fields on the graph, an overlapping traffic area is a feature of a potential anomaly, however, it sometimes does not indicate that the corresponding rules are really overlapping because their fourth field might be different. Nonetheless, PolicyVis still lets users visualize real anomalies by allowing related rules to be investigated more closely. When the user wants to investigate an overlapping area, he simply clicks on it and PolicyVis will focus on more details of the related rules.

PolicyVis first collects all rules containing the selected area, and then sketches a different graph for these rules. In order to correctly view real anomalies with only three fields used on the graph, PolicyVis needs to choose a left-out field which is the same for all the related rules. This common field is guaranteed to exist because related rules from an overlapping area must have at least two fields in common. PolicyVis selects the most common and least important field to be the left-out one if there are multiple common fields among the related rules.

Moreover, among three fields used for the focusing graph, PolicyVis picks the most common field over the related rules to be the third coordinate (the one integrated into visualization objects), and chooses the other two fields as the graph normal coordinates (used for axes). This coordinate selection technique assures users that, from this focusing view, an overlapping area definitely indicates at least one anomaly in the policy.

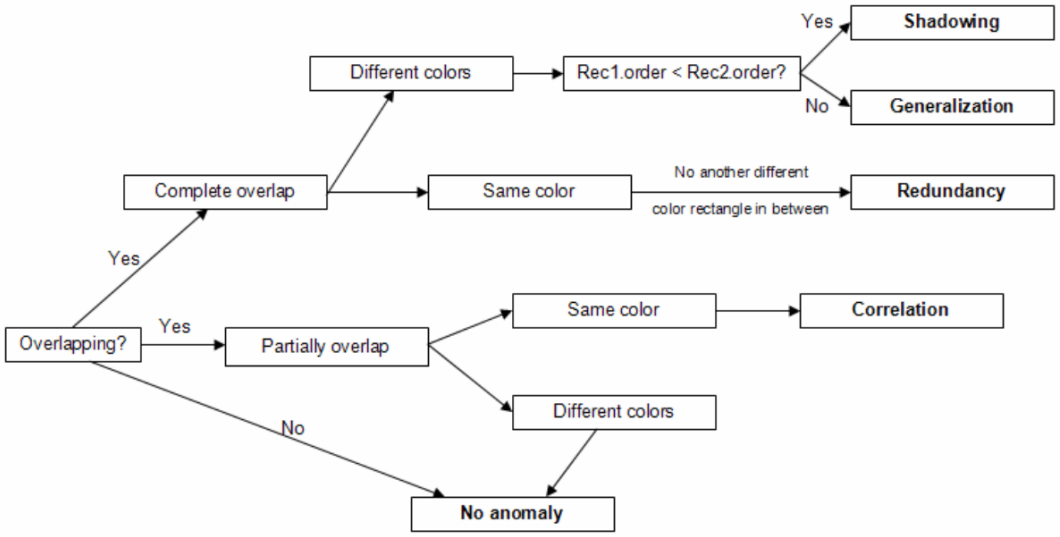


Figure 6: Diagram to determine possible anomalies.

Order	Protocol	SrcIP	SrcPort	DestIP	DestPort	Action
1	tcp	140.192.37.2	*	161.120.33.*	20	ACCEPT
2	tcp	140.192.37.*	*	161.120.33.42	20	DENY
3	tcp	140.192.37.*	*	161.120.33.41	25	ACCEPT
4	tcp	140.192.37.1	*	161.120.33.41	25	DENY
5	tcp	140.192.37.3	*	161.120.33.43	21	ACCEPT
6	tcp	140.192.37.*	*	161.120.33.43	21	DENY
7	tcp	140.192.37.*	*	161.120.33.44	53	ACCEPT
8	tcp	140.192.37.4	*	161.120.33.44	53	ACCEPT
9	tcp	140.192.37.5	*	161.120.33.44	23	ACCEPT
10	tcp	140.192.37.5	*	161.120.33.44	23	DENY
11	tcp	140.192.37.5	34	161.120.33.45	22	ACCEPT
12	tcp	140.192.37.*	35	161.120.33.45	22	DENY
13	tcp	140.192.37.1	*	161.120.33.42	20	DENY
14	udp	*.*.*.*	30	161.120.33.43	50	ACCEPT
15	udp	140.192.37.*	30	161.120.33.43	50	DENY

Figure 7: An example of a firewall policy.

To find the most common field over some firewall rules, for each rule field excluding the Action field, PolicyVis needs to find a rule's field value which is a subset of all other rules' field values, and compute the number of rules that have the field value equal to that rule's field value. The field that has the biggest number is the most common field over the rules. The algorithm FindMostCommonField to find the most common field is implemented as shown in Table 2.

How To Recognize Anomalies In PolicyVis

The rules order is an important factor in understating the policy semantic and determining the firewall anomaly types, especially between shadowing and generalization anomalies. Besides allowing users to see the rules order by moving the mouse over the overlapping area, PolicyVis also uses surrounding rectangles (not color-filled) around the overlapping rule rectangles only in the focusing view to visualize rules or rectangles order in each overlapping area. The width (and height) difference between a rule rectangle and its surrounding one in an overlapping area is called boarder and it basically shows the rule order: the rule or rectangle with bigger boarder comes first in the policy. This technique will offer an easy way to determine the type of the anomaly visually and without any manual investigation.

Shadowing and generalization anomalies: These two anomalies can be recognized by a rectangle totally contained in another rectangle but have different colors (different filtering actions), the rules order (based on extra rectangles) will decide which anomaly the overlapping area belongs to.

Redundancy anomaly: The features used to recognize this anomaly are almost the same as features used to pinpoint shadowing and generalization anomalies. Instead of having different colors, the overlapping rectangles should have the same color (same filtering action) and there is no another different color rectangle appears between them.

Correlation anomaly: This anomaly is corresponding to two rectangles with different colors partially contained in each other.

If two rectangles are not overlapping, there is no anomaly between two rules represented by those two rectangles. With the help of PolicyVis, it is straightforward to pinpoint all anomalies that might exist in the firewall policy. Figure 6 summarizes the method to

Algorithm FindMostCommonField
Input: *rules*
Output: *the most common field among the input rules*

```
1:  for each field in rule.fields/{action}
2:      if field = dest_ip or field = src_ip
3:          C_field = *.*.*.*
4:      end if
5:      if field = dest_port or field = src_port
6:          C_field = *
7:      end if
8:      for each rule R_i in rules { find a field value which is a subset of all other field values }
9:          C_field = C_field ∩ R_i.field
10:     end for
11:     N_field = 0
12:     for each rule i in rules { count the number of rules having field value equal to the common subset value}
13:         if R_i.field = C_field
14:             N_field = N_field + 1
15:         end if
16:     end for
17: end for
18: N = max(N_dest_ip, N_scr_ip, N_dest_port, N_scr_port) { choose the most common field }
19: if N = N_src_port
20:     return src_port
21: end if
22: if N = N_dest_port
23:     return dest_port
24: end if
25: if N = N_src_ip
26:     return src_ip
27: end if
28: if N = N_dest_ip
29:     return dest_ip
30: end if
```

Table 2: Algorithm to find the most common field.

determine different rule anomalies which is very effective in a visualized environment like PolicyVis.

A Case Study

Using PolicyVis to investigate the firewall policy shown in Figure 7, the firewall rules are visualized as shown in Figure 8. The admin sees many overlapping areas which might contain potential rule anomalies. There are five suspected overlapping areas (numbered on the graph) which the user believes contain rule anomalies. From this view only, he suspects that:

1. potential of shadowing anomaly
2. potential of generalization anomaly
3. potential of correlation anomaly
4. potential of redundancy anomaly
5. potential of generalization anomaly

However, in order to make sure that those anomalies are real anomalies, the admin needs to closely investigate each overlapping area. To do this, the admin simply clicks on each selected overlapping area and PolicyVis will focus on and show a more elaborated view for that area.

Shadowing anomaly visualization: When the admin clicks on the overlapping area number *1* (Figure 8), he is brought to the view where all traffic has the same Destination IP address *161.120.33.41* as shown in Figure 9. From this view, it is clear that there

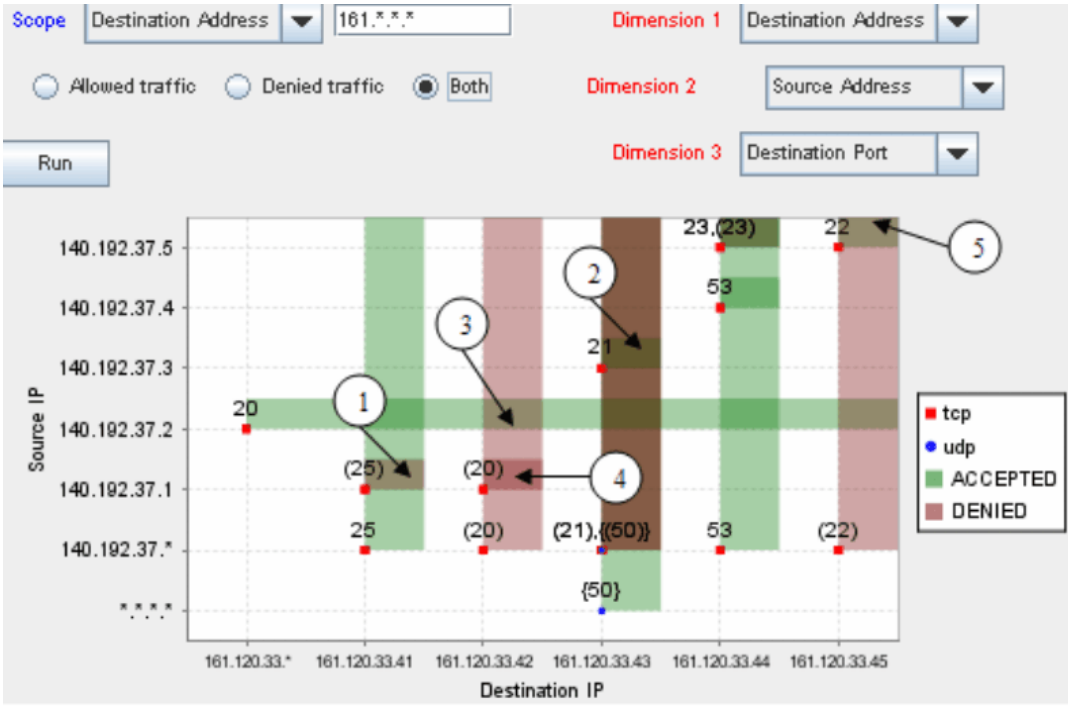


Figure 8: Many potential anomalies in the policy.

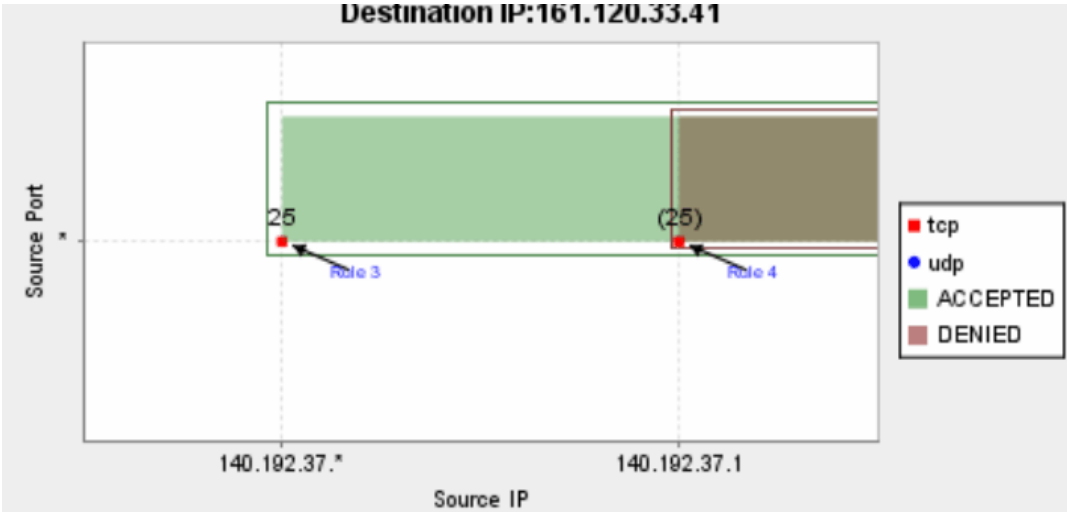


Figure 9: Shadowing anomaly between rule 3 and rule 4.

is a shadowing anomaly between rule *3* and rule *4* (rule *4* is shadowed by rule *3*) because the rectangle representing rule *4* is totally contained in the rectangle representing rule *3* and they have different colors. "Rule 3" and "Rule 4" tooltips appear in this case because the admin moves the mouse over the overlapping area. Without these tooltips, the admin still can tell that this is a shadowing anomaly because he knows the outer rectangle comes first in the policy based on the surrounding rectangles.

Generalization anomaly visualization: When the admin clicks on the overlapping area number *2* (Figure 8), he is brought to the view where all traffic has the same Destination IP address *161.120.33.43* as shown in Figure 10. From this view, it is clear that there is a generalization anomaly between rule *5* and rule *6* (rule *6* is a generalization of rule *5*) because the rectangle representing rule *5* is totally contained in the rectangle representing rule *6* and they have different colors. Moreover, without the tooltips ("Rule 5" and "Rule 6"), the admin still can tell that the inner rectangle comes first in the policy based on the surrounding rectangles and hence this is a generalization anomaly.

Correlation anomaly visualization: When the admin clicks on the overlapping area number *3* (Figure 8), he is brought to the view where all traffic has the same Destination Port *20* as shown in Figure 11. From this view, it is clear that there is a correlation anomaly between rule *1* and rule *2* because the rectangle representing rule *1* is partially overlapped with the rectangle representing rule *2* and they have different colors.

Redundancy anomaly visualization: When the admin clicks on the overlapping area number *4* (Figure 8), he is brought to the view where all traffic has the same Destination IP address *161.120.33.43* as shown in Figure 12. From this view, it is clear that there is a redundancy anomaly between rule *2* and rule 13 (rule 13 is redundant to rule *2*) because the rectangle representing rule *13* is totally contained in the rectangle representing rule *2* and they have the same color.

Overlap but no anomaly: When the admin clicks on the overlapping area number *5* (Figure 8), he is brought to the view where all traffic has the same Destination IP address *161.120.33.45* as shown in Figure 13. From this view, it is clear that there is no anomaly because the rectangles representing rules are

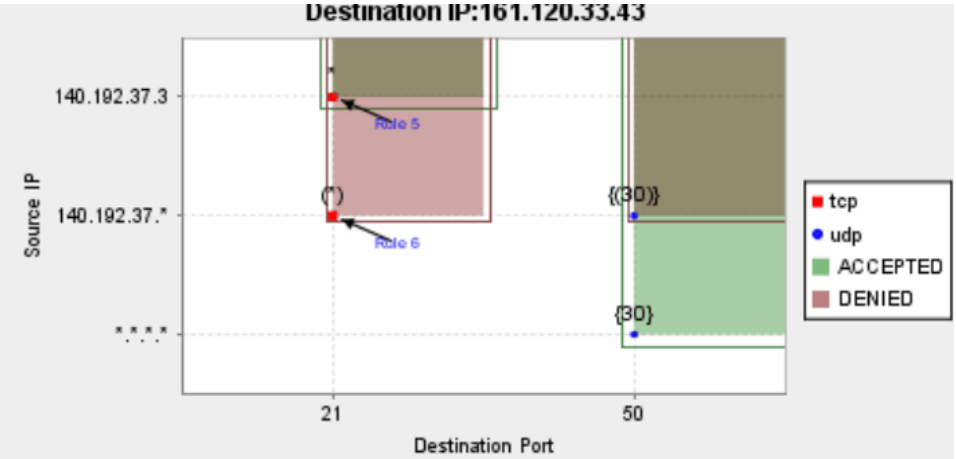


Figure 10: Generalization anomaly between rule 5 and rule 6.

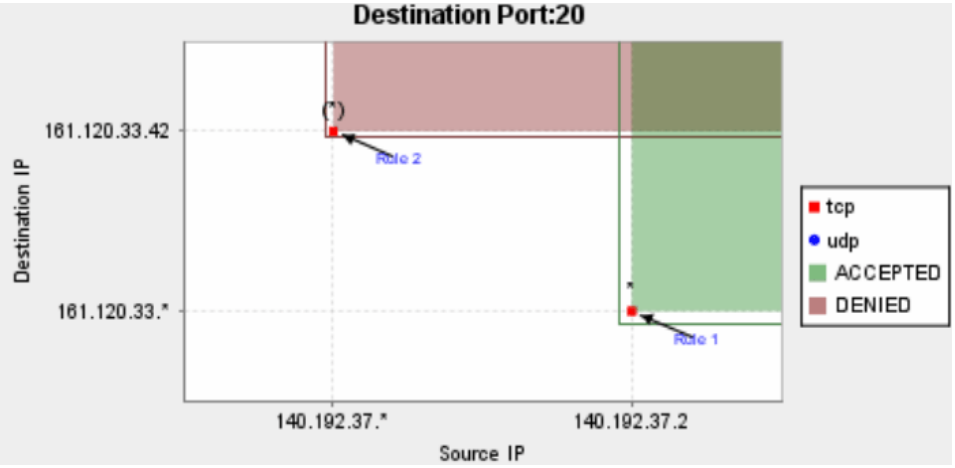


Figure 11: Correlation anomaly between rule 1 and rule 2.

not overlapping. Rule *11* and Rule *12* are overlapped in Figure 8 because Rule *11*'s Destination Address and Source Address are subsets of Rule 12's Destination Address and Source Address respectively and those two fields with Destination Port are chosen as dimensions for the view as shown in Figure 8. However, Rule *11* and Rule *12* have different Source Ports which is automatically chosen by PolicyVis as one of the dimensions for the new view as shown in Figure 13.

Visualizing Distributed Policy Configuration

Concept

While a single firewall is normally deployed to protect a single subnet or domain, distributed firewalls are essential for protecting the entire network. Any misconfiguration or conflict between distributed firewalls might cause serious flaws or damages to the network [2].

Anomalies exist not only in a single firewall but also in inter-firewalls if any two firewalls on a network path take different filtering actions on the same traffic. It is always a higher chance that distributed firewalls contain rule anomalies than a single firewall because of the decentralized property in distributed firewalls management. It is possible that each single firewall in the network might not contain any rule anomaly, but there are still anomalies between different firewalls.

Visualizing distributed firewalls gives the same benefits as visualizing single firewalls in achieving policy behavior discovery, policy correctness checking and anomaly finding. Distributed firewalls are considered as a tree where the root is the borderline firewall which directly filters traffic in and out of the network. Each node in the tree represents for a single firewall which can be placed between subnets or domains in the network.

A packet from outside of the network in order to get through a firewall needs to pass all filterings of all firewalls from the root to the node representing that firewall. In the distributed firewalls view, PolicyVis creates a firewall tree based on the network topology input files and let the user pick a path (from the root to any node) he wants to examine. PolicyVis then builds up a rule set for that path by simply reading rules from nodes in order from the root to the last node. After that, PolicyVis considers this rule set as for a single firewall and visualizes it as before.

A Case Study

The admin wants to investigate the distributed policy configuration applied to traffic to the *Network*

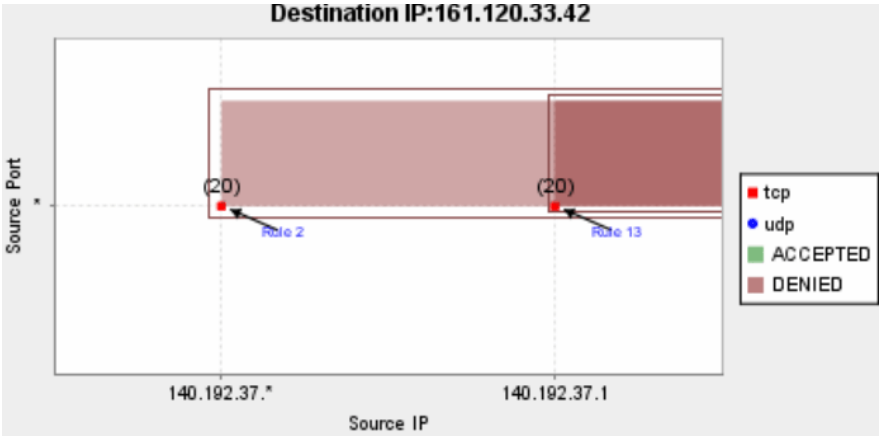


Figure 12: Redundancy anomaly between rule 2 and rule 13.

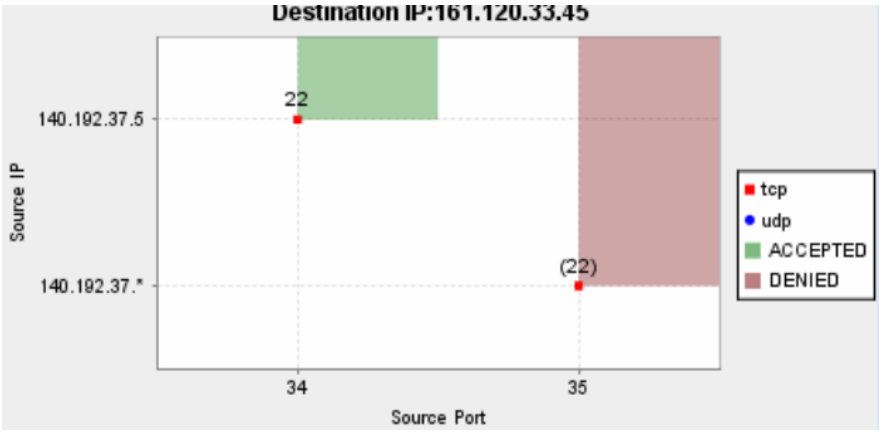


Figure 13: There is no anomaly in this case.

Lab. He first changes the view to *Distributed Firewalls* view and expands the tree to get to the *Network Lab* node. As shown in Figure 14, PolicyVis creates a new rule set containing all rule sets from firewalls on the path in this order: *University of Waterloo*, *Math faculty*, *CS department*, and *Network Lab*.

After building up the rule set for the path from *University of Waterloo* to *Network Lab*, PolicyVis allows the admin to start visualizing the path policy. In this visualization, the admin chooses to investigate all rules on this firewall path that control traffic to any destination address in the university network by choosing the scope *Destination Address* with value *161.*.*.*.

In this case, there are normally multiple subnets get involved because multiple firewalls are considered at once. PolicyVis not only lets the admin visualize all the subnets at the same time, but also supports a single view on each subnet and the admin can switch views between subnets easily. In this example, there are six subnets whose traffic are controlled by the firewalls on the path and the *Network Lab* subnet *161.120.33.** is currently viewed and analyzed by the admin (Figure 15). The admin can change the view to a different subnet by clicking on the *Next* or *Previous* button.

It is easy to recognize that while the single firewall placed at the *Network Lab* subnet (Figure 16) which only controls traffic to *161.120.33.** doesn't contain any anomaly, the distributed firewalls (Figure 15) seems to have anomalies (overlapping areas). In fact, there is a shadowing anomaly in this case between a rule in the *University of Waterloo* firewall and a rule in the *Network Lab* firewall.

Implementation and Evaluation

We implemented PolicyVis using Java and Jfreechart [7], a free open source Java chart library, in PolicyVis to make it easy for displaying charts in the graph. We also used Buddy [17] for BDD representation of firewall policies.

In this section, we present our evaluation study of the usability and efficiency of PolicyVis. To access the practical value of PolicyVis, we not only created firewall policies randomly (with and without rule anomalies), but also used real firewall rules in our

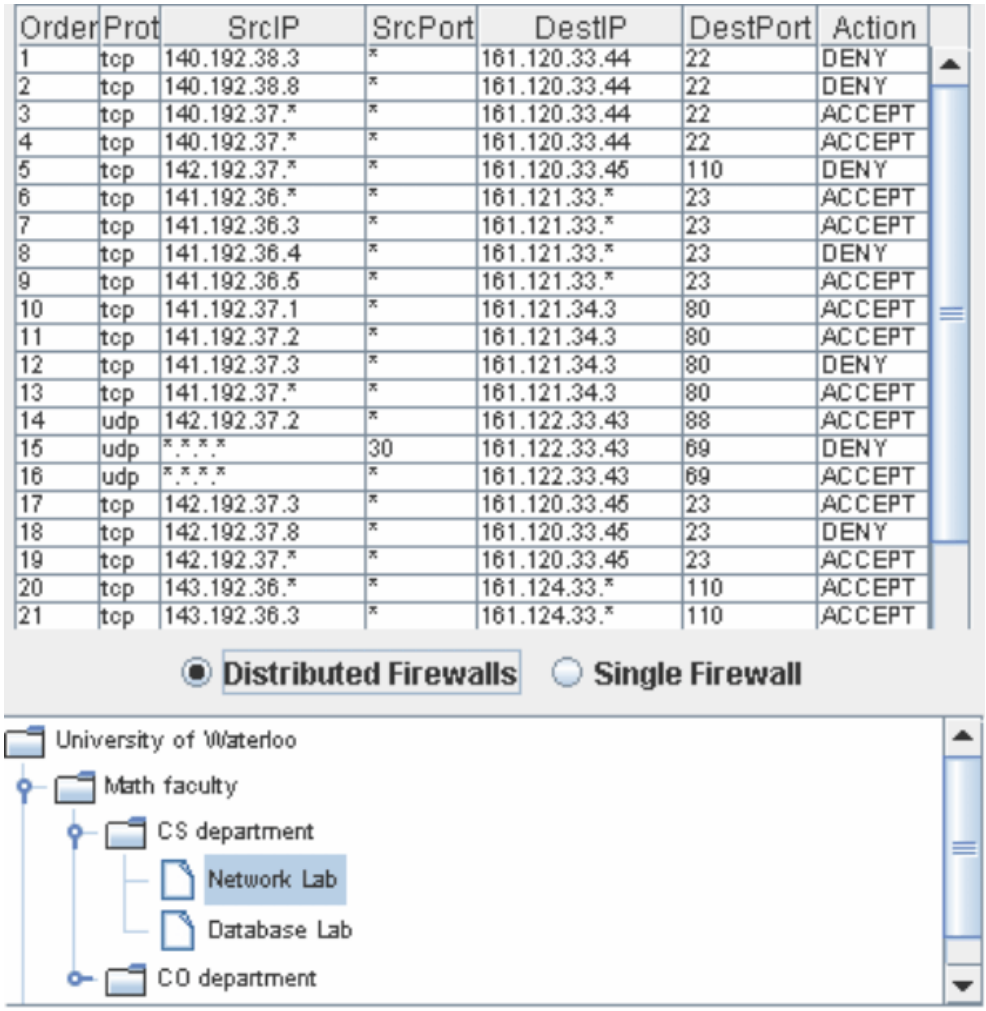


Figure 14: An example of distributed firewalls.

Figure 15: Visualization of all firewall policies to subnet 161.120.33.*.

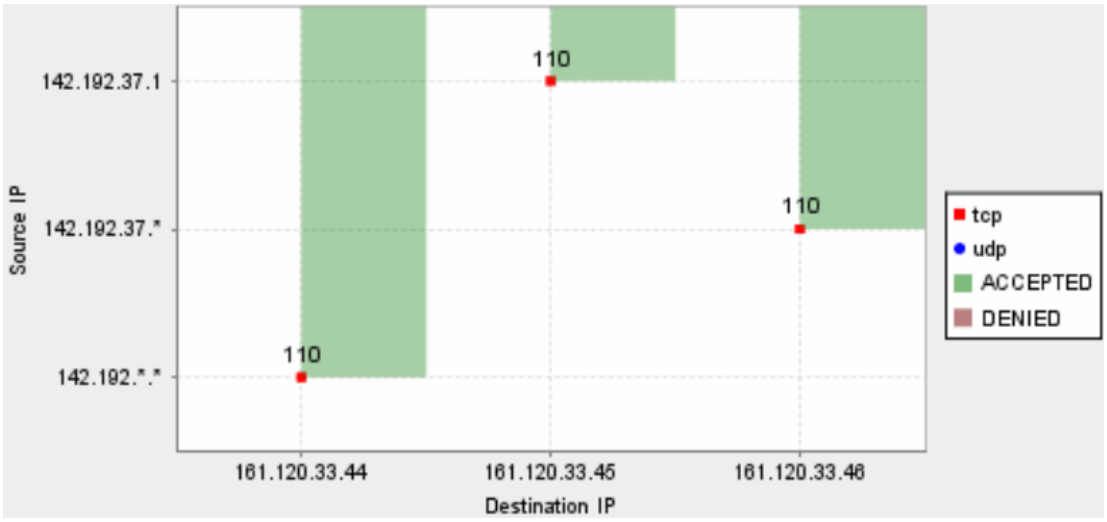


Figure 16: Visualization of the *Network Lab* subnet firewall.

Task\Method	PolicyVis	Raw firewall rules
Find firewall properties	*3.12 minutes*	*10.44 minutes*
Find firewall anomalies	*1.98 minutes*	*12.78 minutes*

Table 3: Average estimated time to achieve each task by using each method.

university for the evaluation study. Each firewall used in the evaluation test has from 30 to 45 rules. We then asked 11 people (with varying level of expertise in the field) under test to use both PolicyVis and raw firewall rules to find some specific firewall properties (like what traffic is allowed to a chosen domain or which machine has Web accessible web traffic and so on) and locate rule anomalies in the firewalls. We recorded the time to answer each task by using each method for all people and computed the average time over all.

People in this evaluation test were getting familiar with PolicyVis very quickly and very confident with features supported by PolicyVis. As shown in Table 3, the average time to achieve each task by using PolicyVis is much faster than by investigating raw firewall rules, especially in finding firewall anomalies. This evaluation test demonstrated that PolicyVis is a very user-friendly tool with high usability and efficiency.

Conclusion and Future Work

Firewalls provide proper security services if they are correctly configured and efficiently managed. Firewall policies used in enterprise networks are getting more complex as the number of firewall rules and devices becomes larger. As a result, there is a high demand for an effective policy management tool which significantly helps user in discovering firewall policy's properties and finding rule anomalies in both single and distributed firewalls.

PolicyVis presented in this paper provides visual views on firewall policies and rules which gives users a powerful means for inspecting firewall policies. In this paper, we described design features of PolicyVis tool and illustrated PolicyVis with multiple examples showing the effectiveness and usefulness of PolicyVis in determining the policy behavior in various case studies. We presented concepts and techniques to find rule anomalies in PolicyVis. Besides, we also showed how PolicyVis visualizes distributed firewalls to achieve same benefits as visualizing single firewalls. Finally, we presented the implementation and evaluation of PolicyVis.

Using PolicyVis was shown very effective for firewalls in real-life networks. In regards to usability, unskilled people with short time of learning how to use PolicyVis can quickly understand and start using all features of PolicyVis. Moreover, by evaluation, PolicyVis effectively helped users including network security juniors and seniors to figure out firewall policy behavior easily by reviewing the visualizing of primitive firewall rules. In addition, PolicyVis was shown a very good tool in finding rule anomalies or conflicts easily and quickly. The number of dimensions users need to consider during firewall inspection varies according to situations; however, considering all possible rule fields always gives users a better analysis of the policy.

Even though PolicyVis was shown a very effective tool, we still want to perform more evaluation on it and collect more users' ideas to make PolicyVis the best visualization tool for firewall policies. There are still many possible features that we want to implement in PolicyVis to maximize its usability as well as efficiency. We want PolicyVis to support more viewing levels of firewall policies and automatically show users possible strange behaviors and true rule anomalies of firewall policies on the graph. In addition, PolicyVis currently shows how to visualize stateless firewalls but we can easily envision extending this to visualize stateful firewalls too by preprocessing the policy to create the implicit rules in stateful firewalls. We consider supporting stateful firewalls in PolicyVis in our future work.

Author Biographies

Tung Tran received the Bachelor of Math with Distinction on the Dean's Honours List from the University of Waterloo in 2006. He is currently a Master student of Computer Science at the University of Waterloo, under the co-supervision of Prof. Raouf Boutaba and Prof. Ehab Al-Shaer. His main research concentrates on network security, especially firewall and IDS policy management. He can be reached at t3tran@cs.uwaterloo.ca .

Ehab Al-Shaer is an Associate Professor and the Director of the Multimedia Networking Research Lab (MNLAB) in the School of Computer Science, Telecommunications and Information System at DePaul University since 1998. Prof. Al-Shaer is also a Research Scientist faculty at School of Computer Science, University of Waterloo.

His primary research areas are Network Security, fault and Configuration management. He is Co-Editor of number of books in the area of multimedia management and end-to-end monitoring. Prof. Al-Shaer has serves as a Program Co-chair for number of well-established conferences in area including IM 2007, POLICY 2008, MMNS 2001, and E2EMON 2003-2005. Prof. Al-Shaer was a Guest Editor for number of Journals in his area. He has also served as TPC member, session chair, and tutorial presenter for many IEEE/ACM/IFIP conferences in his area including INFOCOM, ICNP and IM/NOMS and others. His research is funded by NSF, Cisco, Intel, and Sun Microsystems. Prof. Al-Shaer has received several best paper awards and other awards such as a NASA fellowship. He received his M.S. and Ph.D. in Computer Science from Northeastern University, Boston, MA and Old Dominion University, Norfolk, VA in 1994 and 1998 respectively.

Raouf Boutaba received the M.Sc. and Ph.D. Degrees in Computer Science from the University Pierre & Marie Curie, Paris, in 1990 and 1994 respectively. He is currently a Professor of Computer Science at the University of Waterloo. His research interests include network, resource and service management in multimedia wired and wireless networks.

Dr. Boutaba is the founder and Editor-in-Chief of the IEEE Transactions on Network and Service Management and on the editorial boards of several other journals. He is currently a distinguished lecturer of the IEEE Communications Society, the chairman of the IEEE Technical Committee on Information Infrastructure and the IFIP Working Group 6.6 on Network and Distributed Systems Management. He has received several best paper awards and other recognitions such as the Premier's research excellence award.

References

[1] Al-Shaer, E., H. Hamed, and R. Boutaba, M. Hasan, ''Conflict Classification and Analysis of Distributed Firewall Policies,'' *IEEE Journal of Selected Areas of Communications (JSAC)*, 2005.

[2] Al-Shaer, E. and Hazem Hamed, ''Discovery of Policy Anomalies in Distributed Firewalls,'' *Proceedings of IEEE INFOCOM'04*, March, 2004.

[3] Becker, R. A., S. G. Eick, and A. R. Wilks, ''Visualizing Network Data,'' *IEEE Transactions on Visualization and Computer Graphics*, pp. 16-28, 1995.

[4] Bethel, E. W., S. Campbell, E. Dart, K. Stockinger, and K. Wu, ''Accelerating Network Traffic Analysis Using Query-Driven Visualization,'' *IEEE Symposium on Visual Analytics Science and Technology*, IEEE Computer Society Press, 2006.

[5] El-Atawy, A., K. Ibrahim, H. Hamed, and E. Al-Shaer, ''Policy Segmentation for Intelligent Firewall Testing,'' *1st Workshop on Secure Network Protocols (NPSec 2005)*, November, 2005.

[6] Eppstein, D. and S. Muthukrishnan, ''Internet Packet Filter Management and Rectangle Geometry.'' *Proceedings of 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January, 2001.

[7] Gilbert, David and Thomas Morgner, *JFreeChart, Java Chart Library,* http://www.jfree.org/jfreechart/ .

[8] Goodall, John, Wayne Lutters, Penny Rheingans, and Anita Komlodi, ''Preserving the Big Picture: Visual Network Traffic Analysis with TNV,'' *Proceedings of the 2005 Workshop on Visualization for Computer Security,* pp. 47-54, October, 2005.

[9] Hamed, Hazem, Ehab Al-Shaer and Will Marrero, ''Modeling and Verification of IPSec and VPN Security Policies,'' *Proceedings of IEEE ICNP'2005*, November, 2005.

[10] Hari, B., S. Suri, and G. Parulkar, ''Detecting and Resolving Packet Filter Conflicts.'' *Proceedings of IEEE INFOCOM'00*, March, 2000.

[11] Lakkaraju, K., R. Bearavolu, and W. Yurcik, ''NVisionIP – A Traffic Visualization Tool for Large and Complex Network Systems,'' *International Multiconference on Measurement, Modelling, and Evaluation of Computer-Communication Systems (Performance TOOLS)*, 2003.

[12] Lee, Chris P., Jason Trost, Nicholas Gibbs, Raheem Beyah, John A. Copeland, ''Visual Firewall: Real-time Network Security Monitor,'' *Proceedings of the IEEE Workshops on Visualization for Computer Security*, p. 16, October 26-26, 2005.

[13] Liu, A. X., M. G. Gouda, H. H. Ma, and A. H. Ngu, ''Firewall Queries,'' *Proceedings of the 8th International Conference on Principles of Distributed Systems*, LNCS 3544, T. Higashino Ed., Springer-Verlag, December, 2004.

[14] Le Malécot, E., M. Kohara, Y. Hori, and K. Sakurai, ''Interactively combining 2D and 3D visualization for network traffic monitoring,'' *Proceedings of the 3rd ACM international Workshop on Visualization For Computer Security*, November 3, 2006.

[15] McPherson, J., K. L. Ma, P. Krystosk, T. Bartoletti, and M. Christensen, ''PortVis: A Tool for Port-Based Detection of Security Events,'' *Proceedings of CCS Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC)*, October, 2004.

[16] Nidhi, Sharma, *FireViz: A Personal Firewall Visualizing Tool*, Thesis (M. Eng.), Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2005.

[17] Lind-Nielsen, J., *The Buddy obdd Package,* http://www.bddportal.org/buddy.html .

[18] Wool, A., ''Architecting the Lumeta Firewall Analyzer,'' *Proceedings of 10th USENIX Security Symposium*, August, 2001.

[19] *Tufin SecureTrack: Firewall Operations Management Solution*, http://www.tufin.com .