Firewall Policy Anomaly Management

Deepti N B , Nivedita Annayya Maravante , Rao Nisha A Padmanabh, Sreevalli V
City Engineering College, Doddakalsandra, Bangalore-560061
deepti.nb@gmail.com1 , niveditamaravante@gmail.com2 , raonisha78@gmail.com3 , svalli91@gmail.com4

Abstract

The advent of emerging technologies such as Web services, has enabled us to perform business services more efficiently and effectively. However, we still suffer from unintended security leakages by unauthorized actions in business services. Firewalls are the most widely deployed security mechanism to ensure the security of private networks in most businesses and institutions. Security concerns are becoming increasingly critical in networked systems. Firewalls provide important defense fornetwork security. However, misconfigurations in firewallsare very common and significantly weaken the desired security.The effectiveness of security protection provided by a firewall mainly depends on the quality of policy configured in the firewall. Here an innovative policy anomaly management framework for firewalls, adopting a rule-based segmentation technique to identify policy anomalies and derive effective anomaly resolutions is represented.

Index Terms—*Firewall, policy anomaly management, access control.*

1.Introduction

Firewall is a widely deployed mechanism for improving the security of enterprise networks. However, configuring a firewall is daunting and error-prone even for an experienced administrator. As a result, misconfigurations in firewalls are common and serious. As one of essential elements in network and information system security, firewalls have been widely deployed in defending suspicious traffic and unauthorized access to Internet-based enterprises. Sitting on the border between a private network and the public Internet, a firewall examines all incoming and outgoing packets based on security rules. To implement a security policy in a firewall, system administrators define a set of filtering rules that are derived from the organizational network security requirements.

Firewall policy management is a challenging task due to the complexity and interdependency of policy rules. Corresponding policy analysis tools, such as Firewall Policy Advisor [1] and FIREMAN [5], with the goal of detecting policy anomalies have been introducethe relationships between one rule and the collections of packet spaces derived from all preceding rules. However, each analysis result from FIREMAN can only show that there is a misconfiguration between one rule and its preceding rules, but cannot accurately indicate all rules involved in an anomaly.

Firewall Policy Advisor only has the capability of detecting pairwise anomalies in firewall rules. FIREMAN can detect anomalies among multiple rules by analyzing

On the other hand, due to the complex nature of policy anomalies, system administrators are often faced with a more challenging problem in resolving anomalies, in particular, resolving policy conflicts. An intuitive means for a system administrator to resolve policy conflicts is to remove all conflicts by modifying theconflicting rules. However, changing the conflicting rules is significantly difficult.

In this paper, we represent a novel anomaly management framework for firewalls based on a rule-based segmentation technique to facilitate not only more accurate anomaly detection but also effective anomaly resolution. Based on this technique, a network packet space defined by a firewall policy can be divided into a set of disjoint packet space segments. Each segment associated with a unique set of firewall rules accurately indicates an overlap relation (either conflicting or redundant) among those rules. We also introduce a flexible conflict resolution method to e Enable a fine-grained conflict resolution with the help of several effective resolution strategies with respect to the risk assessment of protected networks and the intention of policy definition.

This paper is organized as follows: Section 2 overviews the anomalies in firewall policies. Section 3 presents an anomaly representation technique based on packet space. In Section 4, we articulate our policy anomaly management framework. In Section 5, we address the implementation details and evaluations of FAME. Section 6 describes the related work. Section 7 concludes this paper.

2. Overview of Anomalies in Firewall Policies

A firewall policy consists of a sequence of rules that define the actions performed on packets that satisfy certain conditions. The rules are specified in the form of <condition, action>. A condition in a rule is composed of a set of fields to identify a certain type of packets matched by this rule. Table 1 shows an example of a firewall policy, which includes five firewall rules r_1, r_2, r_3, r_4, and r_5. Note that the symbol "*" utilized in firewall rules denotes a domain range. For instance, a single "*" appearing in the IP address field represents an IP address range from 0.0.0.0 to 255.255.255.255. Several related work has categorized different types of firewall policy anomalies. [1], [5].

Based on following classification, we articulate the typical firewall policy anomalies

1. Shadowing. A rule can be shadowed by one or a set of preceding rules that match all the packets which also match the shadowed rule, while they perform a different action. In this case, all the packets that one rule intends to deny (accept) can be accepted (denied) by previous rule(s); thus, the shadowed rule will never be taken effect. In Table 1, r_4 is shadowed by r_3 because r_3 allows every TCP packet coming from any port of 10.1.1.* to the port 25 of 192.168.1.*, which is supposed to be denied by r_4.
2. Generalization A rule is a generalization of one or a set of previous rules if a subset of the packets matched by this rule is also matched by the preceding rule(s) but taking a different action. For example, r_5 is a generalization of r_4 in Table 1. These two rules indicate that all the packets from 10.1.1.* are allowed, except TCP packets from 10.1.1.* to the port 25 of 192.168.1.
3. Correlation. One rule is correlated with other rules, if a rule intersects with others but defines a different action. In this case, the packets matched by the intersection of those rules may be permitted by one rule, but denied by others. In Table 1, r_2 correlates with r_5, and all UDP packets coming from any port of 10.1.1.* to the port 53 of 172.32.1.* match the intersection of these rules. Since r_2 is a preceding rule of r_5, every packet within the intersection of these rules is denied by r_2. However, if their positions are swapped, the same packets will be allowed.
4. Redundancy. A rule is redundant if there is another same or more general rule available that has the same effect. For example, r_1 is redundant with respect to r_2 in Table 1, since all UDP packets coming from any port of 10.1.2.* to the port 53 of 172.32.1.* matched with r_1 can match r_2 as well with the same action.

Table 1An Example Firewall Policy

Rule	Protocol	Source IP	Source Port	Destination IP	Destination Port	Actior
r_1	UDP	10.1.2.*	*	172.32.1.*	53	deny
r_2	UDP	10.1.*.*	*	172.32.1.*	53	deny
r_3	TCP	10.1.*.*	*	192.168.*.*	25	allow
r_4	TCP	10.1.1.*	*	192.168.1.*	25	deny
r_5	*	10.1.1.*	*	*	*	allow

Existing conflict classification and detection approaches only treat a policy conflict as an inconsistent relation between one rule and other rules. Given a more general definition on policy conflict as shown in Definition 1, we believe that identifying policy conflicts should always consider a firewall policy as a whole piece, and precise indication of the conflicting sections caused by a set of overlapping rules is critical for effectively resolving the conflicts.

Definition 1(Policy Conflict). A policy conflict pc in a firewall F is associated with a unique set of conflicting firewall rules cr={r_1,..,r_n}which can derive a common network packet space. All packets within this space can match exactly the same set of firewall rules, where at least two rules have different actions: Allow and Deny.
Definition 2(rule redundancy). A rule r is redundant in a firewall F iff the network packet space derived from the resulting policy F'after removing r is equivalent to network space definedby F. That is, F and F' satisfy following equations: $S^A_F = S^A_{F'}$ and $S^D_F = S^D_{F'}$, where S^A and S^D denote allowed and denied network packet spaces, respectively.

Algorithm 1: Segment Generation for Network Packet Space of a Set of Rule R: Partition(R)

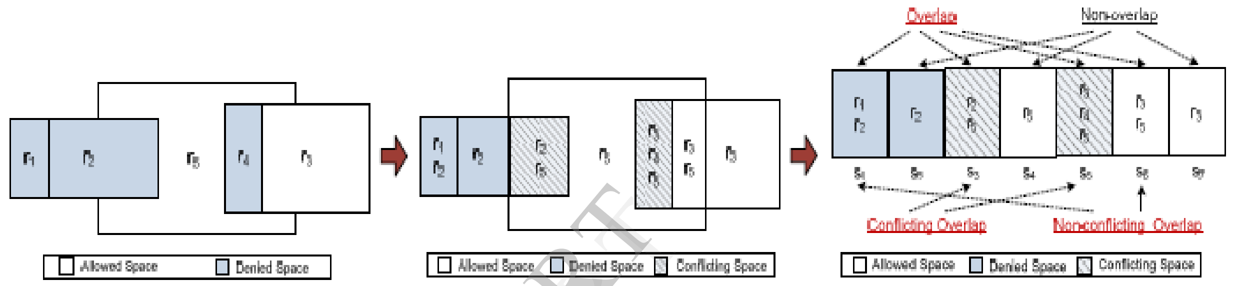
```
Input: A set of rules, $R$.
Output: A set of packet space segments, $S$.
foreach $r \in R$ do
    $s_r \longleftarrow PacketSpace(r)$;
    foreach $s \in S$ do
        /* $s_r$ is a subset of $s$*/
        if $s_r \subset s$ then
            $S.Append(s \setminus s_r)$;
            $s \longleftarrow s_r$;
            $Break$;
        /* $s_r$ is a superset of $s$*/
        else if $s_r \supset s$ then
            $s_r \longleftarrow s_r \setminus s$;
        /* $s_r$ partially matches $s$*/
        else if $s_r \cap s \neq \emptyset$ then
            $S.Append(s \setminus s_r)$;
            $s \longleftarrow s_r \cap s$;
            $s_r \longleftarrow s_r \setminus s$;
    $S.Append(s_r)$;
return $S$;
```

3.Anomaly Representation Based On Packet Space

3.1 Packet Space Segmentation and Classification

In order to precisely identify policy anomalies and enable a more effective anomaly resolution, we introduce a rule-based segmentation technique, which adopts a binary decision diagram (BDD)-based data structure to represent rules and perform various set operations, to convert a list of rules into a set of disjoint network packet spaces. Algorithm 1 shows the pseudo code of generating packet space segments for a set of firewall rules R.2 This algorithm works by adding a network packet space s derived from a rule r to a packet space set S. A pair of packet spaces must satisfy one of the following relations: subset (line 5), superset (line 10), partial match (line 13), or disjoint (line 17). Therefore, one can utilize set operations to separate the overlapped spaces into disjoint spaces.

A firewall rule typically utilizes five fields to define the rule condition; thus, a complete representation of packet space should be multidimensional. Fig. 1a gives the two-dimensional geometric representation of packet spaces derived from the example policy shown in Table 1. We utilize colored rectangles to denote two kinds of packet spaces: allowed space (white color) and denied space (gray color), respectively. In this example, there are two allowed spaces representing rules r_3 and r_5, and three denied spaces depicting rules r_1, r_2, and r_4.



(a) Two dimensional geometric representation of overlapping rules

(b) Packet space segmentation

(c) Uniform representation

Fig. 1 Packet Space Representation Derived From The Example Policy

In order to clearly represent all identical packet spaces derived from a set of overlapping rules, we adopt the rule-based segmentation technique addressed in Algorithm 1 to divide an entire packet space into a set of pairwise disjoint segments. We classify the policy segments as follows: non-overlapping segment and overlapping segment, which is further divided into conflicting overlapping segment and non-conflicting overlapping segment. Each non-overlapping segment associates with one unique rule and each overlapping segment is related to a set of rules, which may conflict with each other (conflicting overlapping segment) or have the same action (non-conflicting overlapping segment). Figure 1(b) demonstrates the segments of packet spaces derived from the example policy. Since the size of segment representation does not give any specific benefits in resolving policy anomalies, we further present a uniform representation of space segments in Figure 1(c). We can notice that seven unique disjoint segments are generated. Three policy segments s2, s4 and s7 are non-overlapping segments. Other policy segments are overlapping segments, including two conflicting overlapping segments s3 and s5, and two non-conflicting overlapping segments s1 and s6.

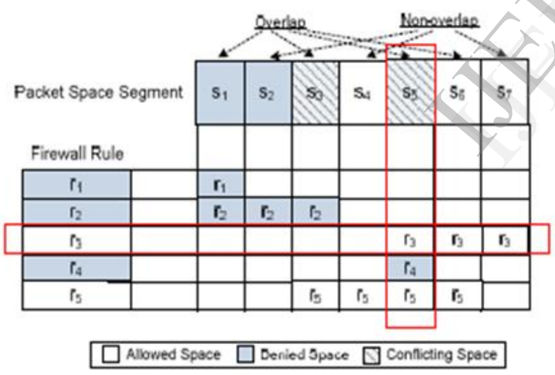


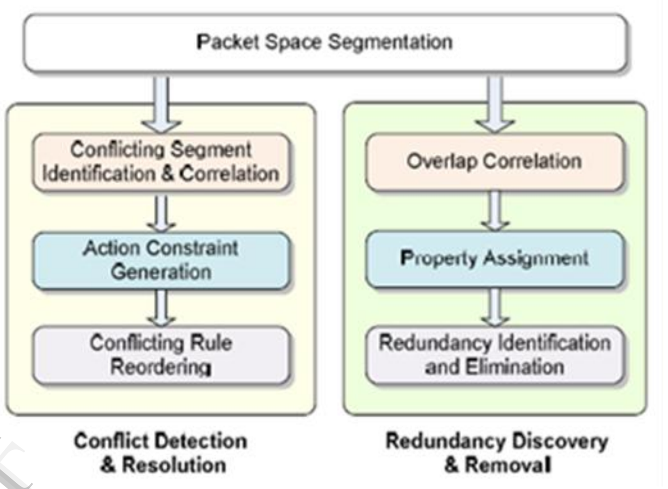
Fig. 2 Grid representation of policy anomaly

3.2 Grid Representation of Policy Anomaly

To address the need of a more precise anomaly representation, we additionally introduce a grid representation that is a matrix-based. visualization of policy anomalies, in which space segments are displayed along the horizontal axis of the matrix, rules are shown along the vertical axis, and the intersection of a segment and a rule is a grid that displays a rule's subspace covered by the segment. Figure 2 shows a grid representation of policy anomalies for our example policy. We can easily determine which rules are covered by a segment, and which segments are associated with a rule.

4.Anomaly Management Framework

Our policy anomaly management framework is composed
of two core functionalities: conflict detection and resolution, and redundancy discovery and removal, as depicted in Figure 3. Both functionalities are based on the rule-based segmentation technique.



Packet Space Segmentation
Conflicting Segment Identification & Correlation
Action Constraint Generation
Conflicting Rule Reordering
Overlap Correlation
Property Assignment
Redundancy Identification and Elimination
Conflict Detection & Resolution
Redundancy Discovery & Removal

Fig. 3. Policy anomaly management framework.

4.1 Correlation of Packet Space Segment

Technically, one rule may get involved in multiple policy anomalies. It is necessary to identify the dependency relationships among packet space segments for efficiently resolving policy anomalies Figure 4 shows an example of segment correlation. 3 Suppose we add three new rules r6, r7 and r8 in the example policy shown in Table 1. Several rules in this firewall policy are involved in multiple.

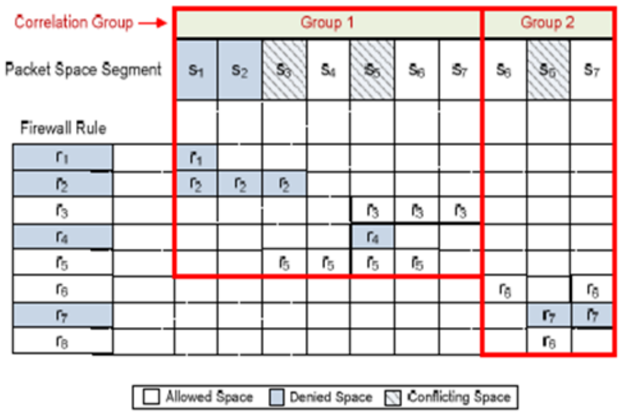


Fig. 4 Example Of Segment Correlation

anomalies. For example, r2 is associated with three segments s1, s2 and s3. Also, we can identify r3, r5, r6 and r7 are also associated with multiple segments. Assume we need to resolve the conflict related to a conflicting segment s3 by reordering associated conflicting rules, r2 and r5. The position change of r2 and r5 would also affect other segments, s1, s2, s4, s5 and s6. Thus, a dependency relationship among those segments can be derived. We cluster such segments with a dependency relationship as a group called correlation group. Consequently, two correlation groups, group1 and group2. group1 contains seven segments and a rule set with five elements (r1, r2, r3, r4 and r5); and group2 includes three segments and three associated rules, r6, r7 and r8. The major benefit of generating correlation groups for the anomaly analysis is that anomalies can be examined within each group independently, because all correlation groups are independent of each other. Especially, the searching space for reordering conflicting rules in conflict resolution can be significantly lessened and the efficiency of resolving conflicts can be greatly improved.

4.2 Conflict Resolution

A practical and effective method to resolve a policy conflict is to determine which rule should take precedence when a network packet is matched by a set of rules involved in the conflict. In order to utilize the existing first-match conflict resolution mechanism implemented in common firewalls, the rule expected to take precedence needs to be moved to the first-match rule. An action constraint is assigned to each conflicting segment. An action constraint for a conflicting segment defines a desired action (either Allow or Deny) that the firewall policy should take when any packet within the conflicting segment comes to the firewall. Then, to resolve a conflict, we only assure that the action taken for each packet within the conflicting segment can satisfy the corresponding action constraint. A key feature of this solution is that we do not need to move a rule expected to take precedence to the first match rule at all times. Any rule associated with the conflict on the same action (as a rule with the precedence) can be moved to the first-match rule, guaranteeing the same effect with respect to the conflict resolution.

4.2.1 Action Constraint Generation Once conflicts in a firewall policy are discovered and conflict correlation groups are identified, the risk assessment for conflicts is performed. A basic idea of automated strategy selection is that a risk level of a conflicting segment is used to directly determine the expected action taken for the network packets in the conflicting segment. If the risk level is very high, the expected action should deny packets considering the protection of network perimeters. On the contrary, if the risk level is quite low, the expected action should allow packets to pass through the firewall so that the availability and usage of network services cannot be affected. Thus, conflict resolution strategies (RS) can be generated automatically for partial conflict segments by comparing the risk levels with two thresholds, upper threshold (UT) and lower threshold (LT), which can be set by system administrators in advance based on the different situations of protected networks.

Risk (security) levels are determined based on the vulnerability assessment of the protected network we adopt the Common Vulnerability Scoring System (CVSS) [15] as an underlying security metrics for risk evaluation. Two major factors, exploitability of vulnerability (reflecting the likelihood of exploitation) and severity of vulnerability (representing the potential damage of exploitation), are utilized to evaluate the risk level of a network system. Beside those two factors, another important factor in determining the criticality of an identified security problem is asset importance value.

Since the CVSS base score can cover both exploitability of vulnerability and severity of vulnerability factors, we incorporate the CVSS base score and asset importance value to compute the risk value for each vulnerability as follows:

Risk Value = (CVSS Base Score) × (Importance Value) (1)

To calculate the risk level (RL) of each conflicting segment, we accumulate all risk values of the vulnerabilities covered by a conflicting segment.

$$RL(cs) = \frac{\Sigma_{v \in V(cs)}(CVSS(v) \times IV(s))}{\alpha \times |V(cs)|} \quad (2)$$

Where, V (cs) is a function to return all vulnerabilities that are contained in a conflicting segment cs; CV SS(v) is a function to return the CVSS base score of vulnerability v; and IV (s) is a function to return the importance value (with the range from 0.0 to 1.0) of service s. Also, we incorporate a coefficient factor ($\alpha \times 1 / |V(cs)| \leqslant \alpha \leqslant 1$) that allows system administrators to express their preferences in choosing average or overall risk value to measure the risk of each conflicting segment. The value of can be decreased, in which case an administrator cares more about the overall risk value. Otherwise, s/he can increase the value of until it reaches an average risk value.We classify conflict resolution strategies for firewall policies into two categories: network situation-aware strategy and policy oriented strategy.

[Network situation-aware strategy]: A system administrator
adopts this strategy to resolve policy conflicts based on the results of risk assessment of the network covered by corresponding conflicting segments.

• Deny-overrides: This strategy indicates that “deny” rules take precedence over “allow” rules. In general, a system administrator may directly take this strategy to harden his network if the risk level of a conflict is very high.

• Allow-overrides: This strategy states that “allow” rules take precedence over “deny” rules. A system administrator may apply this strategy to resolve a conflict, which has a lower risk level.

[Policy-oriented strategy]: This strategy considers the factors related to the policy definition, such as when was the rule defined? what was the rule defined for? and who defined the rule?

• Recency-overrides: This strategy indicates that rules take precedence over rules specified earlier. As the security requirements may change over a period of time, an administrator may define new rules along with his evolving security requirements which may be in conflict with previous security requirements. Obviously, in this case, newer rules should take precedence over older rules.

• Specificity-overrides: This strategy states that a more specific rule overrides more general rules. In a firewall policy, shadowing and generalization conflicts can be identified by conflict detection tools. In our solution, we treat shadowing and generalization conflicts as the same case, since we resolve them through rule reordering.

• High-majority-overrides: This strategy allows (denies, resp.) a packet if the number of rules taking “allow” (“deny,” resp.) action is greater than the number of rules taking “allow” (“deny,” resp.) action.

• First-match-overrides: This strategy states that the first matched rule takes precedence over others. This strategy can be directly converted to the existing firewall implementation.

• High-authority-overrides: This strategy states that a rule defined by an administrator with a higher authority level takes precedence.

4.2.2 Rule Reordering The most ideal solution for conflict resolution is that all action constraints for conflicting segments can be satisfied by reordering conflicting rules. Figure 5 illustrates a scenario representing that action constraints cannot be fully satisfied. In this scenario, four rules intersect with each other in different conflicting segments. The existing order of rules cannot satisfy the fourth action constraint. In order to make the fourth action constraint satisfied, r4 needs to be moved in front of r1. However, this situation causes the violation against the third action constraint. A naive way to find an optimalsolution is to exhaustively search all permutations of correlated conflicting rules. We then compute a resolving score for each permutation by counting how many action constraints can be satisfied, and select the permutation with the maximum resolving score as the best solution for a conflict resolution. However, a key limitation of using the permutation algorithm is its computational complexity which is O(n!). To address this issue, we introduce a greedy algorithm, which can be employed to resolve theconflicts containing a larger number of correlated conflicting rules. A greedy algorithm makes the locally optimal choice at each stage with the hope of finding the global optimum. For all conflicting rules in a correlation group, our greedy conflicting resolution algorithm first calculates a resolving score for each conflicting rule individually. Then, the rule with the greatest resolving score is selected to solve the conflicts: a position range with the best conflict resolution is identified for the selected rule; and moving the selected rule to the new position achieves a locally optimal conflicting resolution. Applying the same processes to the remaining rules recursively until all rules in the correlation group are processed, the final order of the correlated conflicting rules is a solution of the conflict resolution. Note that the resolving scores for the remainder of rules should be recalculated, since moving a rule may change the original conflicting situation of a policy.

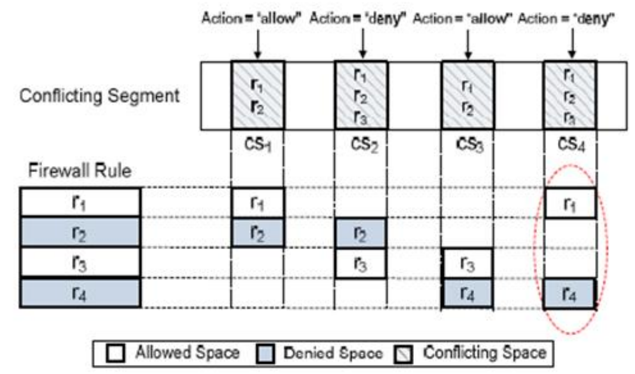


Fig .5Partial Satisfaction Of Action Constraints

4.3 Redundancy Elimination

In this step, every rule subspace covered by a policy segment is assigned with a property. Four property values, removable (R), strong irremovable (SI), weak irremovable (WI) and correlated (C), are defined to reflect different characteristics of each rule subspace. Removable property is used to indicate that a rule subspace is removable. In other words, removing such a rule subspace does not make any impact on the original packet space of an associated policy.

Strong irremovable property means that a rule subspace cannot be removed because the action of corresponding policy segment can be decided only by this rule. Weak irremovable property is assigned to a rule subspace when any subspace belonging to the same rule has strong irremovable property. That means a rule subspace becomes irremovable due to the reason that other portions of this rule cannot be removed. Correlated property is assigned to multiple rule subspaces covered by a policy segment, if the action of this policy segment can be determined by any of these rules. We next introduce three processes to perform the property assignments to all of rule subspaces within the segments of a firewall policy, considering different categories of policy segments.

1) Property assignment for the rule subspace covered by a non-overlapping segment. A non-overlapping segment contains only one rule subspace. Thus, this rule subspace is assigned with strong irremovable property. Other rule subspaces associated with the same rule are assigned with weak irremovable property, except for the rule subspaces that already have strong irremovable property.

2) Property assignment for rule subspaces covered by a conflicting segment. The first rule subspace covered by the conflicting segment is assigned with strong irremovable property. Other rule subspaces in the same segment are assigned with removable property. Meanwhile, other rule subspaces associated with the first rule are assigned with weak irremovable property except for the rule subspaces with strong irremovable property.

3) Property assignment for rule subspaces covered by a non-conflicting overlapping segment. If any rule subspace has been assigned with weak irremovable property, other rule subspaces without any irremovable property are assigned with removable property. Otherwise, all subspaces within the segment are assigned with correlated property.

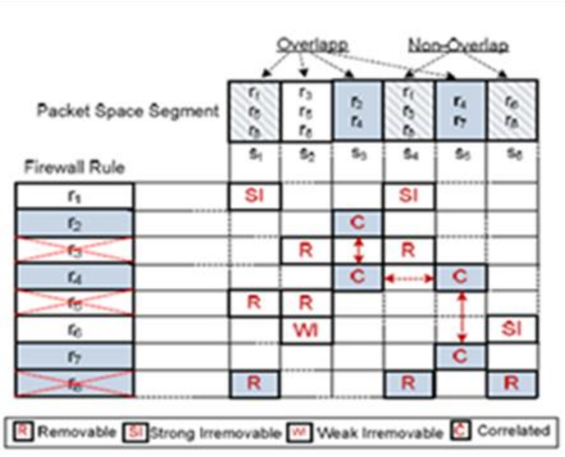


Fig. 6 Example Of Property Assignment.

Figure 6 illustrates the result of applying our property assignment approach, which performs three property assignment processes in sequence, to a firewall policy with eight rules. We can easily identify that three rules, r3, r5 and r8, are removable rules, where the removable property is assigned to all subspaces. In addition, by examining the correlated rules r2, r4 and r7, which contain some subspaces with correlated property, r2 and r7 can be identified as redundant rules and removed from the policy. However, if we leverage traditional redundancy detection method [1], [6], which is limited to detect pairwise redundancies, to this example, only two redundant rules r2 and r7 can be discovered.

5.Related Work

There exist a number of algorithms and tools designed to assist system administrators in managing and analyzing firewall policies. Al-Shaer and Hamed [1] designed a tool called Firewall Policy Advisor to detect pairwise anomalies in firewall rules. Yuan et al. [5] presented FIREMAN, a toolkit to check for misconfigurations in firewall policies through static analysis. As we discussed previously, our framework overcomes the limitations of those tools by conducting a complete anomaly detection and providing more accurate anomaly diagnosis information. In particular, the key distinction is its capability to perform an effective conflict resolution, which has been ruled out in other firewall policy analysis tools.

.

7.Conclusion

In this paper, we have proposed a novel anomaly management framework that facilitates systematic detection and resolution of firewall policy anomalies. A rule-based segmentation mechanism and a grid-based representation technique were introduced to achieve the goal of effective and efficient anomaly analysis. In addition, we have described a proof-of-concept implementation of our anomaly management environment called FAME and demonstrated that our proposed anomaly analysis methodology is practical and helpful for system administrators to enable an assurable network management.

References

[1] E. Al-Shaer and H. Hamed, “Discovery of policy anomalies in distributed firewalls,” in INFOCOM 2004., vol. 4. IEEE, pp. 2605– 2616.

[2] A. Wool, “Trends in Firewall Configuration Errors: Measuring the Holes in Swiss Cheese,” IEEE Internet Computing, vol. 14, no. 4, pp. 58–65, 2010.

[3] J. Alfaro, N. Boulahia-Cuppens, and F. Cuppens, “Complete analysis of configuration rules to guarantee

reliable network security policies," International Journal of Information Security, vol. 7, no. 2, pp. 103–122, 2008.
[4] F. Baboescu and G. Varghese, "Fast and scalable conflict detection for packet classifiers," Computer Networks, vol. 42, no. 6, pp. 717– 735, 2003.
[5] L. Yuan, H. Chen, J. Mai, C. Chuah, Z. Su, P. Mohapatra, and C. Davis, "Fireman: A toolkit for firewall modeling and analysis,"
[6] E. Lupu and M. Sloman, "Conflicts in policy-based distributed systems management," IEEE Transactions on Software Engineering, vol. 25, no. 6, pp. 852–869, 1999.
[7] I. Herman, G. Melanc¸on, and M. Marshall, "Graph visualization and navigation in information visualization: A survey," IEEE Transactions on Visualization and Computer Graphics, pp. 24–43, 2000.
[8] L. Yuan, C. Chuah, and P. Mohapatra, "ProgME: towards programmable network measurement," ACM SIGCOMM Computer Communication Review, vol. 37, no. 4, p. 108, 2007.
[9] A. El-Atawy, K. Ibrahim, H. Hamed, and E. Al-Shaer, "Policy segmentation for intelligent firewall testing," in 1st Workshop on Secure Network Protocols (NPSec 2005), 2005.
[10] G. Misherghi, L. Yuan, Z. Su, C.-N. Chuah, and H. Chen, "A general framework for benchmarking firewall optimization techniques," IEEE Transactions on Network and Service Management, vol. 5, no. 4, pp. 227–238, Dec. 2008.
[11] P. Mell, K. Scarfone, and S. Romanosky, "A complete guide to the common vulnerability scoring system version 2.0," in Published by FIRST-Forum of Incident Response and Security Teams, June, 2007.