Rule Based Segmentation For Firewall Policy Anomalies

P. Shajahan
M.Tech(CSE)
KVSR Engg Colg
Kurnool

V. Trilik Kumar
Assoc. Prof in IT
KVSR Engg Colg
Kurnool

Abstract-The advent of emerging computing technologies such as service-oriented architecture and cloud computing 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. The effectiveness of security protection provided by a firewall mainly depends on the quality of policy configured in the firewall. Unfortunately, designing and managing firewall policies are often error prone due to the complex nature of firewall configurations as well as the lack of systematic analysis mechanisms and tools. In this paper, we represent an innovative policy anomaly management framework for firewalls, adopting a rule-based segmentation technique to identify policy anomalies and derive effective anomaly resolutions. We also discuss a proof-of-concept implementation of a visualization-based firewall policy analysis tool called Firewall Anomaly Management Environment (FAME). In addition, we demonstrate how efficiently our approach can discover and resolve anomalies in firewall policies through rigorous experiments.

1. INTRODUCTION

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. This is further exacerbated by the continuous evolution of network and system environments.

For instance, Al-Shaer and Hamed [1] reported that their firewall policies contain anomalies even though several administrators including nine experts maintained those policies. In addition, Wool [2] recently inspected firewall policies collected from different organizations and indicated that all examined firewall policies have security flaws. The process of configuring a firewall is tedious and error prone. Therefore, effective mechanisms and tools for policy management are crucial to the success of firewalls. Recently,
policy anomaly detection has received a great deal of attention. Corresponding policy analysis tools, such as Firewall Policy Advisor [1] and FIREMAN [5], with the goal of detecting policy anomalies have been introduced. Firewall

Policy Advisor only has the capability of detecting pair wise anomalies in firewall rules. FIREMAN can detect anomalies among multiple rules by analyzing the relationships between one rule and the collections of packet spaces derived from all preceding rules. However, FIREMAN also has limitations in detecting anomalies [3]. For each firewall rule, FIREMAN only examines all preceding rules but ignores all subsequent rules when performing anomaly analysis. In addition, 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. 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 the conflicting rules. However, changing the conflicting rules is significantly difficult, even impossible, in practice from many aspects.

First, the number of conflicts in a firewall is potentially large, since a firewall policy may consist of thousands of rules, which are often logically entangled with each other. Second, policy conflicts are often very complicated. One rule may conflict with multiple other rules, and one conflict may be

associated with several rules. Besides, firewall policies deployed on a network are often maintained by more than one administrator, and an enterprise firewall may contain legacy rules that are designed by different administrators. Thus, without a priori knowledge on the administrators' intentions, changing rules will affect the rules' semantics and may not resolve conflicts correctly.

Furthermore, in some cases, a system administrator may intentionally introduce certain overlaps in firewall rules knowing that only the first rule is important. In reality, this is a commonly used technique to exclude specific parts from a certain action, and the proper use of this technique could result in a fewer number of compact rules [5]. In this case, conflicts are not an error, but intended, which would not be necessary to be changed. Since the policy conflicts in firewalls always exist and are hard to be eliminated, a practical resolution method is to identify which rule involved in a conflict situation should take precedence when multiple conflicting rules (with different actions) can filter a particular network packet simultaneously. To resolve policy conflicts, a firewall typically implements a first-match resolution mechanism based on the order of rules.

In this way, each packet processed by the firewall is mapped to the decision of the first rule that the packet matches. However, applying the first-match strategy to cope with policy conflicts has limitations. When a conflict occurs in a firewall, the existing first matching rule may not be a desired rule that should take precedence with respect to conflict resolution. In particular, the existing first matching rule may perform opposite action to the rule which should be considered to take precedence. This situation can cause severe network breaches such as permitting harmful packets to sneak into a private network, or dropping legal traffic which in turn could encumber the availability and utility of network services.

Obviously, it is necessary to seek a way to bridge a gap between conflict detection and conflict resolution with the first-match mechanism in firewalls. 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 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.

2. RELATED WORK

There exist a number of algorithms and tools designed to assist system administrators in managing and analyzing firewall policies. Lumeta and Fang allow user queries for the purpose of analysis and management of firewall policies. Essentially, they introduced lightweight firewall testing tools but could not provide a comprehensive examination of policy misconfigurations.

There are several interfaces that have been developed to assist users in creating and manipulating security policies. Expandable Grid is a tool for viewing and authoring access control policies. The representation in Expandable Grids is a matrix with subjects shown along the rows, resources shown along the columns, and effective accesses for the combinations of subjects and resources in the matrix cells.

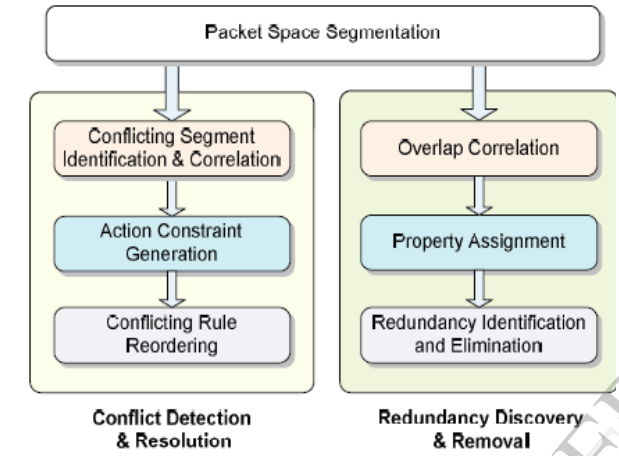
2.1 Firewall Anomalies

Our framework is realized as a proof-of-concept prototype called Firewall Anomaly Management Environment. It is a high-level architecture of FAME with two levels. The upper level is the visualization layer, which visualizes the results of policy anomaly analysis to system administrators. Two visualization interfaces, policy conflict viewer and policy redundancy viewer, are designed to manage policy conflicts and redundancies, respectively. The lower level of the architecture provides underlying functionalities addressed in our policy anomaly management framework and relevant resources including rule information, strategy repository, network asset information, and vulnerability information.

2.2 Anomaly Management Framework in FAME

FAME was implemented in Java. Based on our policy anomaly management framework, it consists of six components: segmentation module, correlation module, risk assessment module, action constraint generation module, rule reordering module, and property assignment module.The segmentation module takes firewall policies as an input and identifies the packet space segments by partitioning the packet space into disjoint subspaces. FAME provides two policy viewers to visualize the outputs of policy conflict analysis and policy

redundancy analysis. Each viewer offers two kinds of visualization interfaces: one interface shows an entire snapshot of all anomalies; another interface shows a partial snapshot only containing anomalies within one correlation group. From below figure depicts interfaces of FAME conflict viewer. The grid representation shows accurately how a set of rules interacts with each other. FAME conflict viewer has the ability to show an overview of the entire conflicts as well as
portions of the policy conflicts, that need to be examined in depth for conflict resolution, based on correlation groups.



Policy anomaly management framework.

Fig 1: Visualization Interfaces in FAME

As illustrated in Fig. 2a, all conflicting segments and conflict correlation groups are displayed along the horizontal axis at the top of the interface. All conflicting rules are shown along the vertical axis at the left of the interface. Each grid cell represents a rule's subspace. In our interface, the icons for conflicting segments indicate four different states with respect to conflicting resolution. One icon represents a conflicting segment with the state of strategy unassigned. Two other icons indicate conflicting segments with the state of strategy assigned with "Allow" action constraint and strategy assigned with "Deny" action constraint, respectively. The fourth icon indicates a conflicting segment with the state of conflict unresolved. In addition, this interface allows an administrator to set the risk level thresholds for automatically assigning strategies. Clicking on a group name box of the interface in Fig 2a, another window as shown in Fig 2b is displayed with the targeted conflicts that an administrator needs to examine and resolve. In this interface, the number of visible entities is reduced to only display conflicting segments in one correlation group and a list of conflicting rules associated with this group. This significantly eliminates administrators' workloads in resolving conflicts by highlighting conflicts within a group. For resolution strategy selection, the administrator needs to further examine rule information for selecting suitable strategies for each conflicting segment. When the administrator clicks the icon of a conflicting segment, the detailed information related to the conflict is displayed in a window as shown in Fig 2c.

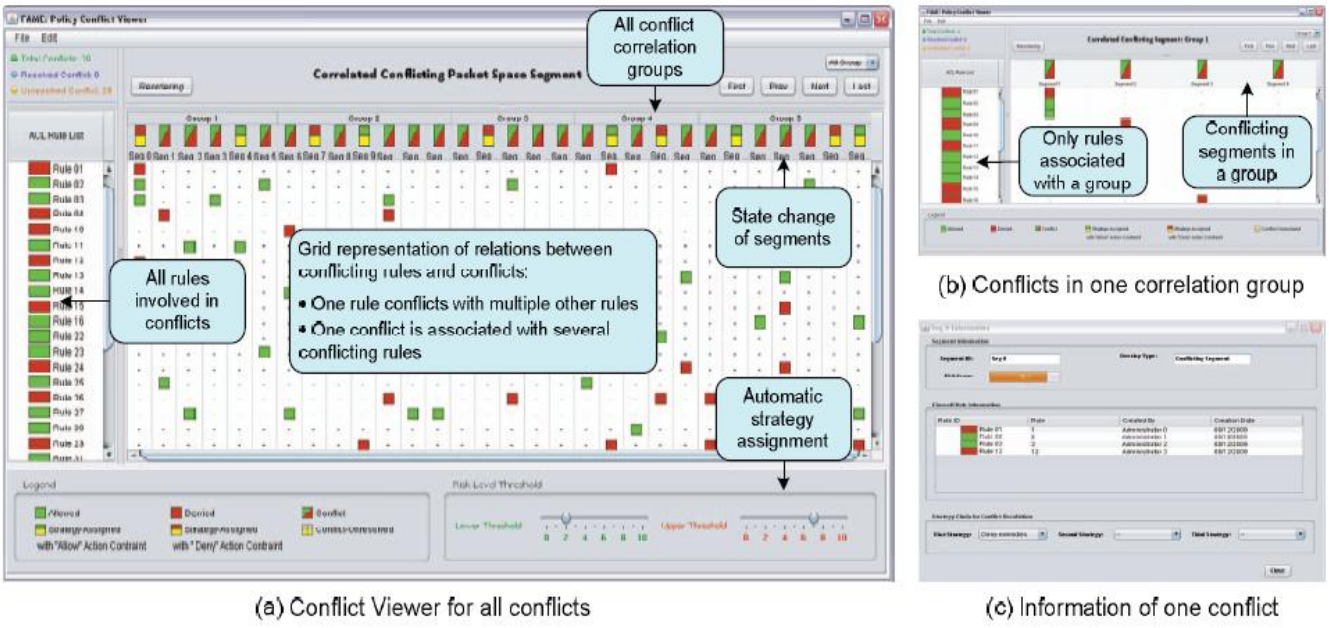
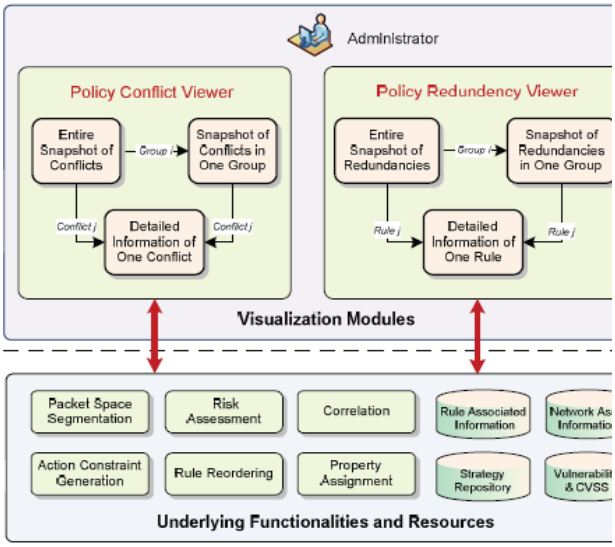


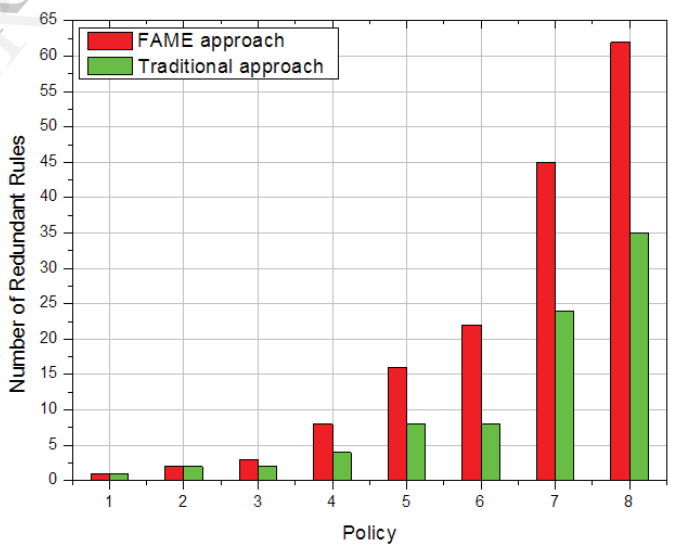
Fig 2 Interface of FAME conflict viewer

2.3 Evaluation of FAME

For FAME evaluation, we utilized a number of firewall policies and associated information required by our tool from different resources. Most of them are from campus networks and some are from major ISPs. Our experiments were performed on Intel Core 2 Duo CPU 3.00 GHz with 3.25 GB RAM running on Linux kernel 2.6.16.



2.4 Result of Analysis



Evaluation of Redundancy Removal

1. We also evaluated our redundancy analysis based on those experimental firewall policies.

2. We observed that FAME could identify an average of 6.5 percent redundancy rules from the whole rules.

3. However, traditional redundancy analysis approach could only detect an average 3.8 percent of total rules as redundant rules

Architecture of FAME

4. 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 assumable network management. Our future work includes usability studies to evaluate functionalities and system requirements of our policy visualization approach with subject matter experts. Also, we would like to extend our anomaly analysis approach to handle distributed firewalls. Moreover, we would explore how our anomaly management framework and visualization approach can be applied to other types of access control policies.

5. REFERENCES

[1] E. Al-Shaer and H. Hamed, "Discovery of Policy Anomalies in Distributed Firewalls," IEEE INFOCOM '04, vol. 4, pp. 2605-2616, 2004.

[2] A. Wool, "Trends in Firewall Configuration Errors: Measuring the Holes in Swiss Cheese," IEEE Internet Computing, vol. 14, no. 4, pp. 58-65, July/Aug. 2010.

[3] J. Alfaro, N. Boulahia-Cuppens, and F. Cuppens, "Complete Analysis of Configuration Rules to Guarantee Reliable Network Security Policies," Int'l J. 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," Proc. IEEE Symp. Security and Privacy, p. 15, 2006.

[6] A. Wool, "Architecting the Lumeta Firewall Analyzer," Proc.10th Conf. USENIX Security Symp., vol. 10, p. 7, 2001.

[7] A. Mayer, A. Wool, and E. Ziskind, "Fang: A Firewall Analysis Engine," Proc. IEEE Symp. Security and Privacy, pp. 177-189, 2000.