International Journal of Computer Science and Informatics

Volume 4 | Issue 2 Article 8

October 2014

MANAGING REDUNDANCY AND CONFLICTS IN DISTRIBUTED FIREWALLS

MS. KRUTHI K. KUMAR
Dept of CSE, CMRIT, Bangalore, India, kruthi.kumar@gmail.com

MANAGING REDUNDANCY AND CONFLICTS IN DISTRIBUTED FIREWALLS

MS. KRUTHI K. KUMAR[1], MRS. SHANTHI M.B.[2], DR. JITENDRANATH MUNGARA[3]

[1]M.Tech, [2]Asst. Prof, Dept of CSE, CMRIT, Bangalore, India
[3]Prof & Dean, Dept of CSE & ISE, CMRIT, Bangalore, India

Abstract- The use of firewall has been widespread in all the emerging technologies such as Service Oriented Architecture, web services, cloud computing and so on. The term security itself is the most important task that has to be maintained in the real-time applications. Policies are enrolled in the security of the firewall where the quality of policies is to be maintained. The network administrator defines the policy as a rule. Managing the firewall policies, maintaining the risk analysis and also the conflicting nature that arise in the network, lack of systematic analysis mechanisms and tools used are often error prone. The distributed firewall is used to overcome the shortcomings of the traditional firewall. In this paper we represent a set of techniques such as, rule-based segmentation technique to identify the policy anomalies and effectively derive the anomaly resolution. Grid-based visualization technique, provide the policy anomaly information in a grid form, which helps in identifying the policy conflicts and finally the techniques to resolve the conflicts and the redundancy that arise in a single- or multi-firewall environment. We also discuss about the implementation of the visualization-based firewall policy analysis tool called Firewall Anomaly Management Framework (FAME), where all the techniques are used in a single tool and an approach to resolve the anomalies in an effective and efficient way.

Keywords- *Firewall, distributed, access control, policy anomaly management, FAME tool.*

I. INTRODUCTION

A firewall can either be software- based or hardware-based and is used to help keep a network secure. Its primary objective is to control all the incoming and outgoing network traffic by analyzing the data packets and determining whether it should be allowed to pass through or not. Before implementing security policy in a firewall, the system administrators should define the set of filtering rules and also that rule should be derived from the organizational network security requirements. As configuring a firewall is tedious and error prone, few mechanisms and tools that are used should be identified and analysed before using it. The corresponding tools such as firewall policy advisor [1] and FIREMAN [2] are used to detect the policy anomalies. In firewall policy advisor, it detects only the pair wise anomalies in firewall rules. FIREMAN [2], anomalies is detected among the multiple rules where by analysing relationship between one rule and the set of packet spaces is derived from the preceding rules. There is a limitation in FIREMAN that is, it examines all the preceding rules but it ignores the subsequent rules. So that the result shows, there is a misconfiguration of one rule and the preceding rule and cannot directly specify that all the rules are involved in an anomaly. Due to the complex nature of the policy anomalies, the system administrator should face many problems such as, resolving anomalies, removing redundancy, and policy conflicts. Removing the conflicts and modifying the conflicting rules is difficult because, the number of conflicts in the firewall is too large, since it contains thousands of rules and often overlaps with one another. The policy conflicts are often complicated. The priori knowledge should be there for the administrator about the rules, where by changing rules will affects rule semantics and the conflicts cannot be resolved. The system administrator thinks that first rule is important. By allowing the first rule to pass through, if it is trusted that is, if the rule doesn't contain any conflicts or redundancy, then it is allowed, but later leaving all the preceding rules to pass through. This is the main drawback of the first-match strategy. If without checking the preceding rules, and if that rule may contain conflicts or redundancy rule and allowed to pass through, then the policy conflicts cannot be resolved.

In this paper, we use the tool called as Firewall Anomaly Management Environment (FAME). In this tool, the technique such as rule- based segmentation and grid- based visualization techniques are used. Where, the policy conflicting rules and also the conflict redundancy rules will be resolved. In rule-based segmentation technique, the network packet space is derived based on the policies that are divided into a set of disjoint packet space segments. In these segments, a unique set of firewall rules accurately indicates overlap relation (either conflicting or redundancy) among those rules. These overlapping of rules can be analysed and identified by using set operation. The grid- based visualization approach represents the policy anomaly in an effective way by using the matrix- based representation, where the individual rules can be identified.

II. RELATED WORK

The area of firewall and policy- based management has done a significant amount of work. We focus with

the study of number of algorithms and tools designed by the system administrator. Bellovin et al. [3] introduced a distributed firewall model in which a centralised policy specification is supported. Al-Shaer and Hamed [1] designed a tool called as a Firewall policy advisor, where the pairwise anomalies is detected in a firewall rules. Yuan et al. [2] presented FIREMAN, introduced a toolkit in which the misconfigurations are checked in the firewall policies through static analysis. As we have discussed about the FAME tool, that overcomes the limitations of those tools by conducting complete anomaly detection, resolving the conflicts, removing redundancy and also providing accurate anomaly diagnosis information. Using the FAME in distributed, the effective and accurate measure of rules that contains non-conflicting are detected and made free from anomalies. Hari et al. [4] an algorithm is provided for detecting and resolving conflicts in policy filter.

III. OVERVIEW OF TYPICAL FIREWALL POLICY ANOMALIES

A firewall policy consists on number of rules in which sequence of actions performed on the basis of certain conditions. The rules are specified in the form of <condition, action>. In a rule condition is used to identify a certain type of packets matched by the rule. Table 1 shows example of firewall policy which includes rules such as r_1, r_2, r_3, r_4 and r_5. The symbol "*" denotes a domain range. If a single "*" appearing in the IP address field represent the IP address range in the form of 0-255.

The order of rules can be identified as,
<order><protocol><s_ip><s_port><d_ip><d_port><action>

There are different types of firewall policy anomalies. They are classified as,

Shadowing: In this rule, even though they perform different action, the packet is checked for the match with the Preceding rule and also matches with the

TABLE 1: An example of firewall policy

Rule	protocol	Source IP	Source Port	Destination IP	Destination Port	action
r1	TCP	10.1.1.*	*	192.168.*.*	25	Allow
r2	TCP	10.1.*.*	*	192.168.1.*	25	Deny
r3	UDP	10.2.*.*	*	172.32.1.*	53	deny
r4	TCP	10.1.*.*	*	192.168.*.*	25	Allow
r5	*	10.1.*.*	*	*	*	deny

Shadowed rule. In this the performed action is allow or deny. In Table 1 r_2 is shadowed by r_4 because r_4 allows every TCP packet coming from any port of 10.1.1.* to port 25 of 192.168.1.*, in which r_2 is supposed to be denied.

Generalization: in this rule, one or a set of previous rules of a subset of packets matched by this rule is also matched with the preceding rule(s) but taking different action. In Table 1 two rules indicate that all packets from 10.1.1.* are allowed, except with the TCP packet that contains form 10.1.1.* to the port 25 of 192.168.1.*.

Correlation: If a rule is intersected with one another or is overlapped with the other but defines the different action. Then it is said to be correlated with the other. In Table 1 rule r_3 correlates with rule r_5, and all UDP packets coming from port 10.1.1.* 10 port 53 of 172.32.1.* match with the intersection of the other rule. Since r_3 is the preceding rule of r_5, every packet with the intersection of these rule are denied by r2.

Redundancy: A rule is said to be redundant, if a rule contains a same effect that is available with the other rule. In table 1 r_1 is redundant to r_4.

These are the types of anomalies which has to be identified and resolved.

Definition 1: (Policy conflicts). In a firewall F, the policy conflicts pc consists of a unique set of conflicting rules cr = $\{r_1, r_2, .., r_n\}$ which derives a common network packet space. In this space, all the packets match exactly the same firewall rules and perform two actions ALLOW and DENY.

Definition 2: (rule redundancy). A rule r is said to be redundant in firewall F, iff the network packet space R is derived from the resulting policy F' after removing r is same as the network packet space defined by F. $FR^A = F'R^D$ where R^A and R^D are allow and deny of the network packet space.

IV. FUNCTIONALITIES INTEGRATED IN FAME

The FAME (Firewall Anomaly Management Environment) is implemented in JAVA. In this tool the techniques such as rule- based segmentation and grid- based visualization techniques are involved and are used in a single framework called FAME. The anomalies such as conflicting and redundancy are identified and resolved on the basis of this technique.

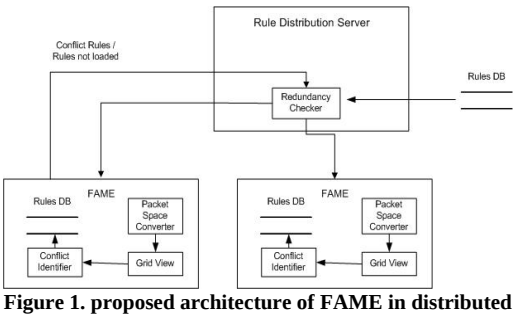

Figure 1. proposed architecture of FAME in distributed firewall.

Figure 1 consists of FAME framework. Using packet converter, the space and the capacity of the packets are identified and are segmented to know which rule is present in that segment. Figure 2 shows the step by step procedure of identifying the anomalies. The rule-based segmentation technique is used, which adapts the binary decision diagram (BDD) based on the data structure that represents the rule and performs the set operations in identifying the overlapping and non-overlapping of rules. The rules are identified either of the subset, superset or disjoint set. Non-overlapping consists of unique rule but overlapping consists of set of rules.

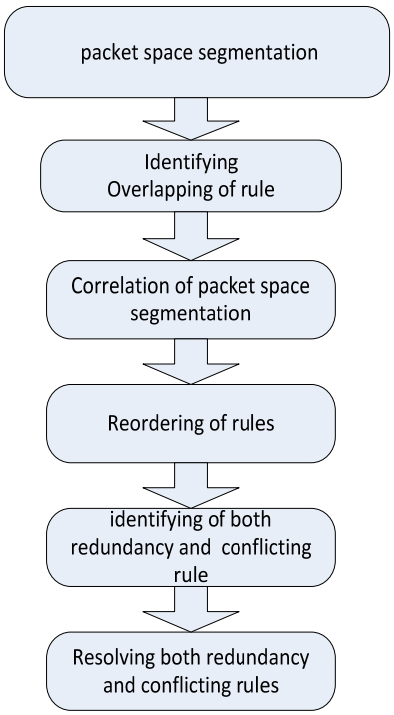

Figure 2. Identifying and resolving policy anomalies.

The overlapping of rules is clearly identified using the technique called grid- based visualization technique shown in figure 2. Grid- based representation technique consists of 'n' number of rules which contains overlapping and non-overlapping conflicting rules. It is represented in the form of matrix, where the overlapping, conflicting rules can be identified easily (rule r_1, r_2 and r_3 in the packet space segment s_3).

In figure 3, the rule r_1 and r_2 are overlapped in the packet segments. The rule r_1, r_3 and r_4 are unique in the packet space segments s_1, s_4, s_6, and s_7 respectively. Rules are shown along the vertical axis and packet space segments along the horizontal axis in the matrix. Intersection of rule and the segment is grid form, which displays the subspaces in which the rule is covered by the segments. In this method, identifying the conflicting rule is easy, by making out which rule is covered by segment and which segment associated with what rule. In this, first step is to identify the conflicting rules; second step is to generate the action constraints for each conflicting segments and third step to reordering of rules.

For example, for only 5 rule if the capacity is much more, then for more than 5 rules it takes many more spaces. Moreover the time consuming will also be more.

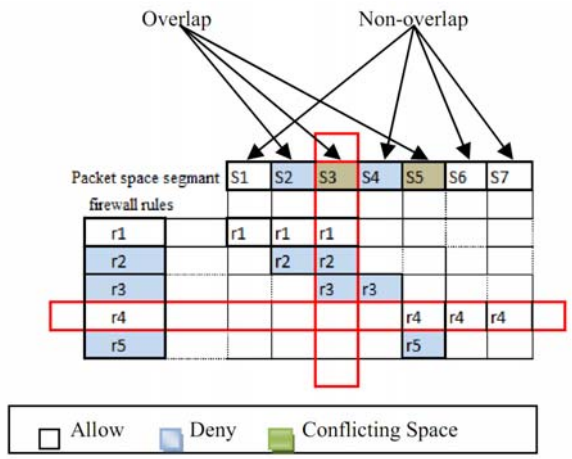

Figure 3. Identifying policy anomaly using grid

In correlation of packet space segments, one rule may be involved in other rule which may cause an unexpected impact on the many other rules. If we suppose to remove one rule then, if that rule is associated with other rule then it is difficult to remove that rule. Similarly, reordering of conflicting rule that is associated with other conflicting rules also cause an impact on the other rules. Therefore identifying the dependency relationship among packet space segments makes it possible to resolve the policy anomalies effectively. In the grid method, the addition of rules may create much space where the identification of rules within the segment becomes more difficult.

For example, if we add two more rules r_6 and r_7 into the figure 3 which consists of the rules associated with the segments containing conflicts and policy anomalies. The rules which are added also overlap with the other rule and conflict will arise. By adding extra rule to the grid, the memory space will be increased and the size will be more.

The risk level of the conflicts can be utilized in both automated and manual strategy. Based on the vulnerability assessment of the protected network the risk (security) level is determined. The co-efficient factor α ($\frac{1}{|r_i \cap s_j|}$) $\leq \alpha \leq 1$ is used to measure the overall risk of each conflicting segments.

Reordering of the rule is a difficult task because, if the rule is associated with the other rule and the overlapping of rules leads to conflicts. If one rule is said to allow and the other deny, then we have to

remove the deny rule and the remaining rule which has to be sent are allowed. In reordering, the position of the rule in the grid has to be changed. There should be no conflicts within the rules. But reordering may cause an impact on the other rule. This is time consuming and memory space is more.

To overcome this problem hashing technique is introduced. In this technique overlapping and non-overlapping conflicting rules can be identified and resolved. The time consuming and the memory space can be maintained.

Steps that hashing technique consists of:

1. Hashing is done to improve the performance in doing the conflict identification.
2. Using mod operation, we calculate the percentage of conflicts present in the rule.
3. The rules with its packet space are hashed to a binary string.
4. The position range of the binary string is dependent on the width of packet space.
5. Generated hash keys are compared with each other; if there is a match, the corresponding rules are considered as conflicting rules.

Here in hashing technique figure 4, two or more rules are compared with one another on the basis of conflicts and overlaps among the rules. The percentages of less conflicting rules are identified and are hashed and stored in the hash table. The percentage of conflicting rules present can be calculated using mod operation. The rules that are stored in the hash table are matched with the other rules that are to be checked. If there is a match found, then there will be a conflict in rules. So the matched rule is not allowed to enter and is ignored. The conflicting rules arise because one rule is involved with the other. First, need to remove the dependency among the rules i.e., redundancy. Second, conflicts i.e., the rules that are overlapped. After resolving conflicts and redundancy, then it will be sent to store in the hash file. The width of packet space is made minimum such that the position ranges of the binary string are capable to fit into.

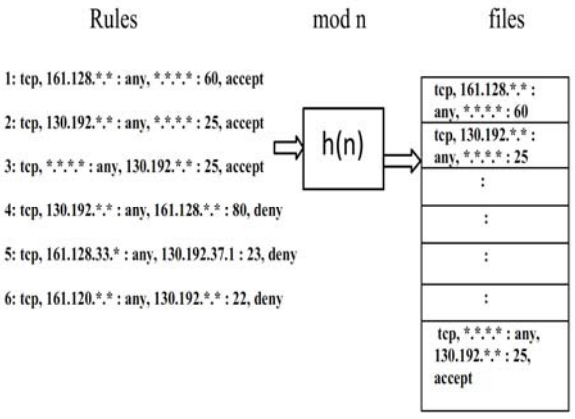

Figure 4: Rules stored in hash table.

Figure 5 shows the screen shots where the splitting of packet space, identifying the conflicts and resolving conflicting rules done in a single framework. The button browse is used to load the rules which is stored in a file. Before putting the rules in a rule configuration file, the rule is first resolved by redundancy i.e., duplicate rules present in a file is removed. Then, ovelapping of rules is identified using packet space segmentation. In this segmentation, the rule can be easily identified by knowing which rule is placed on wich segment and what rule is present on that segment. This method is done for identifying overlapping of rules. The corelation of packet segments is used to identify the rules.

Figure 5. All the method used to resolve conflicts used in a single framework.

In this method, one rule is involved in other rules and removing one rule causes an impact on the other rule. This arise a conficts between the rules and also in reordering of rules, the conflicting rules may be involed with the other rules. This also causes an impact on the other rule. So, before removing conflicting rules and reordering of rules, the dependent rule should be identified and removed. So that, the independent rules are identified and can be allowed to pass through and that rule is stored in a file.

Figure 6 shows the resolved conflicting rules. In this, the rule r_4 was overlapped with the rule r_3 . The dependent rule is identified and allow rule is seperated from the deny rule. The dependency between the rules are resolved. After removing the dependency, the rules are identified as independent. The rule is said to allow if, that rule doesnot matches with rule that is present in the hash file. If there is no match found then it is allowed or else denyed.

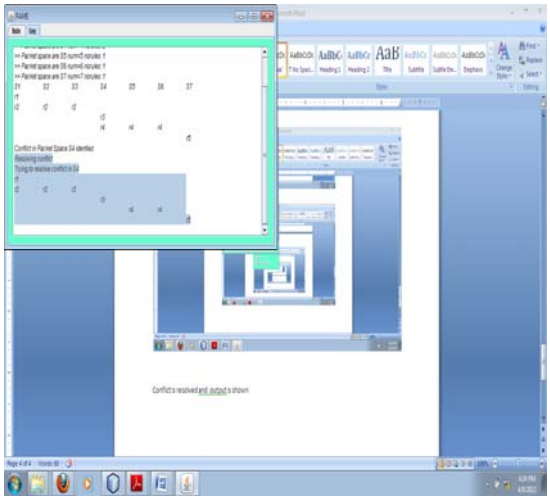

Figure 6. resolved conflicts

Figure 7 shows the performance time to find the conflicts in the distributed FAME using hashing technique. In FAME approach, the time used to store the rules, in identifying the conflicts and reordering of rules consumes more time. If we are going to add new rules then the memory space will be made more to store the rules. But in hashing technique of the distributed FAME approach, the time consumption is much more less and memory space can be maintained.

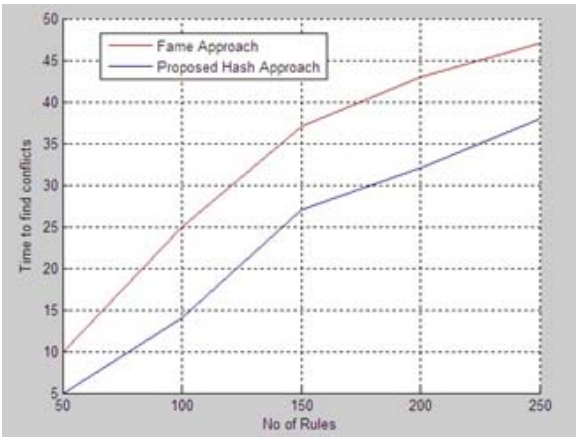

Figure 7.processing time to find conflicts.

V. CONCLUSION

In this paper, the techniques such as rule-based and grid-based are implemented in a single framework called FAME used in a distributed firewall. In this FAME tool all the anomaly conflicting and redundancy rules are identified and resolved in a single framework using distributed firewall. Security of the information is maintained. The memory space utilised is less and less time consuming. The system administrator should have complete knowledge of network management because security plays an important role in the real time applications. Here in this paper, the basic SHA1 hash function is used. Future enhancement is focused on following function. Qos with respect to time complexity and space complexity can be studied using HMAC hashing algorithm which acts as a black box where, it can embed any advanced hashing function. Time and space efficiency in removing conflicts and redundancy using HMAC if, studied and compared with the same in SHA1, which is proposed in the current technique.

REFERENCE

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

[2] L. Yuan, H. Chen, J. Mai, C. Chuah, Z. Su, P. Mohapatra, and C. Davis, "Fireman: A toolkit for firewall modeling and analysis," in 2006 IEEE Symposium on Security and Privacy, 2006, p. 15.

[3] S. Ioannidis, A. Keromytis, S. Bellovin, and J. Smith, "Implementing a distributed firewall," in Proceedings of the 7th ACM conference on Computer and communications security. ACM, 2000, p. 199.

[4] A. Hari, S. Suri, and G. Parulkar, "Detecting and resolving packet filter conflicts," in IEEE INFOCOM, 2000, pp. 1203–1212.

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

[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] 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.

[8] A. Mayer, A. Wool, and E. Ziskind, "Fang: A firewall analysis engine," in IEEE Symposium on Security and Privacy. IEEE Computer Society, 2000, pp. 177–189.

[9] 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.

[10] F. Baboescu and G. Varghese, "Fast and scalable conflict detection for packet classifiers," Computer Networks, vol. 42, no. 6, pp. 717– 735, 2003.

[11] R. Reeder, L. Bauer, L. Cranor, M. Reiter, K. Bacon, K. How, and H. Strong, "Expandable grids for visualizing and authoring computer security policies," in Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems. ACM, 2008, pp. 1473–1482.

[12] M. Gouda and X. Liu, "Firewall Design: Consistency, Completeness, and Compactness," in Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04). IEEE Computer Society, 2004, p. 327.

[13] 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.

[14] A. Wool, "Architecting the lumeta firewall analyzer," in Proceedings of the 10th conference on USENIX Security Symposium-Volume 10. USENIX Association, 2001, p. 7.

[15] J. D. Guttman, "Filtering postures: Local enforcement for global policies", Proceedings of the IEEE Symposium on

Security and Privacy. IEEE, Oakland, CA, pp. 120-129, 1997.

[16] M. E. Whitman, "Enemy at the gate: threats to information security", Communication of the ACM Vol. 46, No. 8, August 2003, pp. 91-95.

[17] S. Cobb. "ICSA Firewall Policy Guide v2.0." NCSA Security White Paper Series, 1997.

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

[19] A. Liu, F. Chen, J. Hwang, and T. Xie. Xengine: A fast and scalable xacml policy evaluation engine. ACM SIGMETRICS Performance Evaluation Review, 36(1):265–276, 2008.