Analysis of Policy Anomalies in Distributed Firewalls

Yu-Zhu Cheng[1] and Qiu-ying Shi[2]
(Corresponding author: Yu-Zhu Cheng)
School of Software, Changsha Social Work College[1]
Changsha 410004, China
(Email: vogue21ct@qq.com)
School of Computer Science and Engineering, Central South University[2]
Changsha 410083, China
(Received Oct. 17, 2021; Revised and Accepted Mar. 12, 2022; First Online Apr. 11, 2022)

Abstract

In order to solve the problem of anomaly analysis in distributed firewalls, a rule anomaly detection method based on spatial relationship comparison is proposed. Firstly, all firewall rules on the network path are mapped into the Firewall Design Matrix(FDM) in reverse order to form independent unit space sets. Secondly, the unit space overlap of all the upstream firewalls corresponding to the most downstream firewall FW_d is obtained. Then this overlap is mapped to the rule space of FW_d, where the uncovered area of FW_d is the desired shadowing anomaly. For spuriousness anomaly, we first calculate the unit space overlap of all downstream firewalls corresponding to the most upstream firewall FW_u and then map the overlap to the rule space of FW_u, where the uncovered area of FW_u is the desired spuriousness anomaly. Simulation results show that this method can accurately and efficiently detect all the rule shadowing anomalies and spuriousness anomalies.

Keywords: Anomaly Detecting; Distributed Firewall; Firewall Policy; Policy Anomaly Analysis

1 Introduction

Firewalls are critical components of network security and are deployed at the entrances between a private network and the Internet to monitor all incoming and outgoing packets. The function of a firewall is to examine the field values of every packet and decide whether to accept or discard a packet according to the firewall policies. The policy is specified as a sequence of rules, each of which has a predicate over some packet header fields and a decision to be performed upon the packets that match the predicate. With the rapid development of the Internet, it is more and more difficult to efficiently manage firewall rules as the number of rules increases. It is known that the rules in a firewall policy are logically entangled because of conflicts among rules and the resulting order sensitivity [23]. Ordering the rules correctly in a firewall is critical and difficult.

In a traditional perimeter firewall environment, the local firewall policy may include intra-firewall anomalies, where the same packet may match multiple filtering rules. Moreover, in distributed firewall environment, firewalls might also have inter-firewall anomalies when individual firewalls in the same path perform different filtering actions on the same traffic [1]. Therefore, the administrator must give special attention not only to all rule relations in the same firewall in order to determine the correct rule order, but also to all relations between rules in different firewalls, in order to determine the proper rule placement in the proper firewall. In addition, a typical large-scale enterprise network might involve hundreds of rules that might be written by different administrators at different times. This significantly increases the potential of anomaly occurrence in the firewall policy, jeopardizing the security of the protected network. Therefore, the effectiveness of firewall security depends on the provision of policy analysis techniques that network administrators can use to analyze the correctness of written firewall filtering rules.

In this paper, we address the problem of anomaly analysis in distributed firewalls. Our work presents a significant contribution in this field since it offers a new approach to analyze anomalies within distributed firewall filtering rules, which is based on the complete definition of anomalies stated in the work of Al-Shaer and Hamed [1]. Our paradigm is based on a two-stage analysis process. In the first stage, for intra-firewall anomalies, we design an approach to eliminate them while maintaining the consistency, compactness and completeness of the original firewall rules [10]. In the second stage, we propose a new approach to analyze inter-firewall anomalies based on the comparison of rule spatial relation, this method can ac-

curately and efficiently discover the shadowing anomalies and spuriousness anomalies of distributed firewall policy. The approach is also very effective, performance analysis and simulation results show that the algorithm has a high execution efficiency.

The rest of this paper is organized as follows: the related work is presented in Section 2; then we define the intra-firewall anomalies, and present the algorithm for elimination of them in Section 3. In Section 4, we first give the classification and detection algorithm of inter-firewall anomalies, followed by the analysis of experimental results in Section 5. Finally, the conclusion is drawn in Section 6.

2 Related Work

Although distributed firewall policy analysis has been given strong attention in the research community, some excellent algorithms and corresponding tools are mostly focused on the problems of rule design, rule compression and rule conflict detection for traditional perimeter firewalls [8–10, 14, 16], while methods and tools for anomaly detection of distributed firewall policies are not many yet [17, 18, 22].

Nowadays, the design, optimization and management of firewall policies have attracted wide attention of researchers, the problem of firewall policy design is to design corresponding firewall rules according to the network security requirements described by natural language. At present, the common method is to define and analyze firewall security policies using specific design models, such as Trie binary tree [3], FDD [10], FDM [8] *et al*, and then generate filtering rules through policy mapping. In our prior work [8], an approach to designing firewall based on multidimensional matrix was proposed. Specifically, we developed a new designing model, namely firewall design matrix (FDM), and the corresponding construction algorithm for mapping firewall rules to FDM, whose consistency and compactness can be achieved by the construction algorithm, and then a firewall generation algorithm was proposed to generate the target firewall rules equivalent to the original ones while maintaining the completeness.

During the process of firewall filtering packets, when a packet matches two or more rules at the same time and the decision of these rules differs, rules conflict. At this time, data packets are processed according to the decision defined by the high priority rule. Generally speaking, rule conflicts need to be detected and eliminated as many as possible, otherwise some packets will be filtered incorrectly, which will bring some adverse consequences to the network. At present, most of the research on rule conflict focuses on the traditional perimeter firewall, including conflict classification, conflict detection and conflict elimination.

The classification of rule conflicts have a detailed discussion in [11]. The traditional perimeter firewall rule conflicts are classified as shadowing anomaly, correlation anomaly, generalization anomaly and redundancy anomaly. Rule conflict detection is to find all conflict rule pairs in a rule set, or to find all rules that conflict with a rule set. Conflict elimination technology refers to the rule set does not have any conflict rules after rule conflicts are eliminated. Literature [12] compares and analyzes the inconsistent parts of rule decision-making in each segment to detect rule conflicts and to eliminate them by adjusting the order of rules. But because rules often span multiple segments, it is possible to introduce anomalies to other segments when adjusting the order of rules in one segment. Moreover, in some cases, no matter how to adjust, it can not achieve the purpose of eliminating conflicts [2]. In addition, with the development of network applications, the number of firewall rules is increasing, it is too complicated to detect and eliminate conflicts manually by administrators, even though recent studies have been trying to resolve errors among polices and optimize a performance of firewall [4, 7, 21]. For this reason, it is important to reveal policy conflicts and potential problems automatically, and provide an intuitive solution to the network administrator. F.Chen *et al.* presented a method to automatically detect rule anomalies [6], they first defined a fault model includes five types of faults: wrong order, missing rules, wrong decisions, wrong predicates, and wrong extra rules. For each type of fault, they proposed a correction technique based on the passed and failed tests of a firewall policy,where the passed and failed tests are automated generated and classified based on packet generation and classify techniques [13]. The approach is effective to correct a faulty firewall policy with faults of wrong order, wrong decisions, and wrong extra rules, but it is not ideal for the correction results of two types of faults: missing rules and wrong predicates.

In distributed firewall environment, the local firewall policy may include intra-firewall anomalies, and might also have inter-firewall anomalies when individual firewalls in the same path perform different filtering actions on the same traffic. In [1], Al-Shaer and Hamed classified various intra-firewall and inter-firewall anomalies, prominent of which being Shadowing, Redundancy, Correlation and Generalization. Based on the specified anomalies, a Firewall Policy Advisor tool (FPA) was developed to detect the existing firewall anomalies in the specified network. The disadvantages of the tool include detection of only pair wise firewall anomalies. Similar to FPA, Lihua Yuan *et al.* [24] introduced Fireman, a toolkit for anomaly analysis in distributed firewalls, while it evaluates firewall configurations as a whole piece rather than just limiting to relation between two firewall rules. Chi-Shih Chao proposed an anomaly diagnosis system in which a RAR (Rule Anomaly Relation) tree is created based on ACL's [5]. The system detect inter- as well as intra-firewall anomalies in a feasible time range. Also, the system suggests network administrators regarding correction in behavior mismatching errors. In [19], Pedditi *et al.* presented a design of a new protocol namely FIEP (Firewall Information Exchange Protocol) which provides a communi-

cation mechanism for distributed firewalls to communicate with each other. Like the Border Gateway Protocol (BGP) that enables routers to exchange routing information, the firewalls can automatically check for inconsistencies in their firewall configuration through message passing. The disadvantages of the protocol include the need to change the existing enterprise hardware which is time consuming and expensive, difficult to implement in existing networks. In order to solve the problem of insufficient usability caused by the limitations of text interface and the complexity of practical use, K. Taeyong *et al.* presented a three-dimensional hierarchical visualization method F/Wvis for intuitive ACL management and analysis [15]. F/Wvis can provide in-depth user interface through hierarchical visualization method, support ACL management of large-scale networks and analysis of policy details and exceptions.

In this paper, we will discuss the inter- and intra-firewall anomalies in detail along with the definition of distributed firewall network model in [1], which is the basis of our anomaly analysis of the distributed firewall policies.

3 Analysis and Detection of Intra-firewall Anomalies

3.1 Intra-firewall Anomaly Definition

Packet classification is performed by sequentially matching the packet against firewall rules until a match is found. If the rules are independent of each other, the order between them is inessential. However, it is very common to have firewall rules interrelated while their decisions are different. In this case, the rules in a firewall policy are logically entangled because of conflicts and the resulting order sensitivity.

Therefore, an intra-firewall policy anomaly 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 that cross the firewall [10].

3.2 Intra-firewall Anomaly Classification

According to the definition of policy anomaly, the rule configuration anomaly can be classified as conflict, incomplete and redundancy. Take the firewall which has four rules as an example:

r_1:F_1∈[0,8] ∧ F_2∈[3,7] →*accept*,
r_2:F_1∈[0,9] ∧ F_2∈[3,9] →*discard*,
r_3:F_1∈[2,7] ∧ F_2∈[3,6] →*accept*,
r_4:F_1∈[0,8] ∧ F_2∈[0,3] →*accept*.

Although the number of rules in this firewall is small, it exemplifies all the three problems of firewall, namely consistency, completeness and compactness. Consistency means that the rules are ordered correctly, completeness means that every packet satisfies at least one rule in the firewall, and compactness means that the firewall has no redundant rules [10].

The two rules r_1 and r_2 are conflicting because there are packets whose fields satisfy the predicates of both r_1 and r_2 (for example, a packet with F_1∈[0,8] ∧ F_2∈[3,7] can satisfy the predicates of both r_1 and r_2) and these two rules have different decisions. Therefore, the relative order of these two rules with respect to one another in the sequence of rules becomes very critical. The relative order of rules r_1 and r_2 is likely a consistency error. The second error in the above rule sequence is that any packet with F_1∈[9,9] ∧ F_2∈[0,2] does not satisfy the predicate of any of the four rules. Such an error is referred to as a completeness error, which can be corrected by adding one new rule r_5:F_1∈[0,9] ∧ F_2∈[0,9] →*discard*. The third error in the above rule sequence is that r_3 is redundant. That is to say, if rule r_3 is removed, the effect of the resulting policy will be unchanged. Such an error is referred to as a compactness error.

3.3 Intra-firewall Anomaly Analysis

In [8], we proposed a firewall rule design method based on multidimensional matrix. By mapping the rules into multidimensional matrix in reverse order, the object rules are non-redundant, conflict-free and completed.

For simplicity, we consider a two-dimensional firewall policy which contains six rules,

r_1:F_1∈[2,6] ∧ F_2∈[4,6] →*accept*,
r_2:F_1∈[8,9] ∧ F_2∈[8,9] →*accept*,
r_3:F_1∈[3,5] ∧ F_2∈[0,5] →*accept*,
r_4:F_1∈[7,8] ∧ F_2∈[0,5] →*discard*,
r_5:F_1∈[6,8] ∧ F_2∈[0,5] →*accept*,
r_6:F_1∈[0,8] ∧ F_2∈[0,9] →*discard*.

After mapping these rules into the multidimensional matrix, we obtain three independent unit spaces: [(3,6)(0,6)],[(8,9)(8,9)],[(2,2)(4,6)], the corresponding firewall rules are

r_1:F_1∈[3,6] ∧ F_2∈[0,6] →*accept*,
r_2:F_1∈[8,9] ∧ F_2∈[8,9] →*accept*,
r_3:F_1∈[2,2] ∧ F_2∈[4,6] →*accept*,
r_4:F_1∈[0,9] ∧ F_2∈[0,9] →*discard*.

It can be seen that the semantics of the object rules are the same as the original policy, while the generated rules r_1,r_2,r_3 are independent of each other without any redundancy and conflicts, and r_4 ensures the completeness.

4 Analysis and Detection of Inter-firewall Anomalies

4.1 Inter-firewall Anomaly Definition

In general, an inter-firewall anomaly may exist if any two firewalls on a network path take different filtering actions on the same traffic. Referring to Figure 1, we assume a traffic flowing from domain D_1 to domain D_2. At any

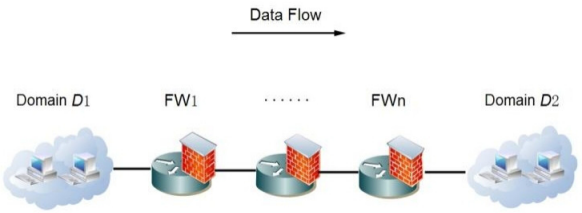


Figure 1: Cascaded firewall isolating domains D_1 and D_2

point on this path in the direction of flow, a preceding firewall is called an upstream firewall, whereas a following firewall is called a downstream firewall. The closest firewall (FW_1) to the flow source domain (D_1) is called the most upstream firewall, while the closest firewall (FW_n) to the flow destination domain (D_2) is called the most downstream firewall.

Even if each firewall policy in the network does not contain the rule anomalies described in Section 3, there could be anomalies between policies of different firewalls. For example, an upstream firewall might block a traffic that is permitted by a downstream firewall or *vice versa*.

As defined in [1], using the network model as shown in Figure 1, for any traffic flowing from domain D_1 to domain D_2, an anomaly exists if one of the following conditions holds:

1) The most downstream firewall accepts a traffic that is blocked by any of the upstream firewalls;

2) The most upstream firewall permits a traffic that is blocked by any of the downstream firewalls;

3) A downstream firewall denies a traffic that is already blocked by the most upstream firewall.

At the same time, all upstream firewalls should permit any traffic that is permitted by the most downstream firewall in order that the flow can reach the destination.

4.2 Inter-firewall Anomaly Classification

In the cascaded firewall network shown in Figure 1, it is assumed that there are no anomalies in the intra-firewall. According to the definition of distributed firewall policy anomaly, if the downstream firewall denies the traffic that has been blocked by the most upstream firewall, a redundancy anomaly will occur. However, we note that according to the definition of intra-firewall anomaly analysis, after eliminating the intra-firewall anomaly, all rules are *accept* except for the last default rule. Considering that the firewall policy follows the principle of "reject everything that is not explicitly allowed", and from the perspective of ensuring the integrity of the rule, we accept the redundancy of this rejection rule. Therefore, for the "condition in definition of the distributed firewall policy anomaly: the downstream firewall denies the traffic blocked by the most upstream firewall.", we do not define this condition as a policy anomaly when detecting the inter-firewall anomalies. In other words, we focus on the following two types of policy anomalies:

1) *Shadowing anomaly* (A_{sh}): the shadowing anomaly occurs if the upstream firewall blocks the network traffic accepted by a downstream firewall;

2) *Spuriousness anomaly* (A_{sp}): the spuriousness anomaly occurs if the upstream firewall permits the network traffic denied by a downstream firewall.

4.3 Inter-firewall Anomaly Detection Algorithm

In this section, we firstly define the related concepts of our approach, and then introduce the algorithm for detecting Inter-firewall anomalies. Table 1 lists the notations used in this article. Taking the distributed firewall network shown in Figure 1 as an example, there are n cascaded firewalls between domains D_1 and D_2, named FW_1,FW_2,...,FW_n respectively. It is assumed that these n firewalls in the network have been processed by the rule mapping method based on multidimensional matrix [8], which means that the rules in each firewall are independent of each other and have decision *accept*, except for the last default *discard* rule. For simplicity, we use blank strips to describe the "*accept*" firewall rule, as shown in Figure 2 and Figure 3. According to the classification of inter-firewall anomalies, in the first case, if an upstream firewall blocks the traffic allowed by the most downstream firewall FW_n, a shadowing anomaly A_{sh} will occur. In another case, a spuriousness anomaly A_{sp} occurs when the most upstream firewall FW_1 allows traffic discarded by a downstream firewall.

Table 1: Notations used in this article

Notation	Paraphrase
F_i	The i^{th} dimension
$D(F_i)$	Domain of F_i
FW_i	The i^{th} Firewall
FW_u	The most upstream firewall
FW_d	The most downstream firewall
R	Firewall rule
US	Unit space
M_k	A k-dimensional matrix
FDM	Firewall design matrix model
A_{sh}	Shadowing anomaly
A_{sp}	Spuriousness anomaly

4.3.1 Shadowing Anomaly Detection

Suppose there are four cascaded firewalls in the network path, named FW_1,FW_2,FW_3 and FW_4 respectively, and

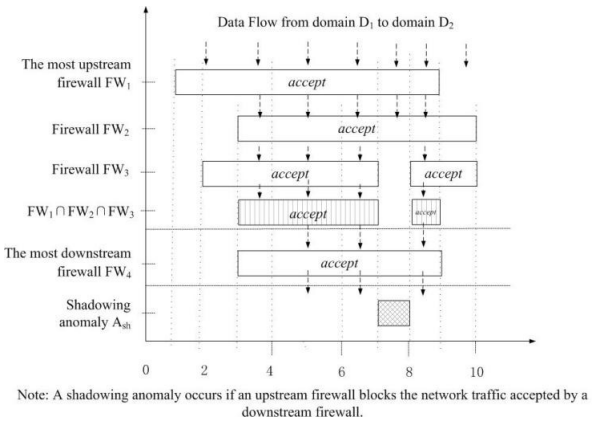


Figure 2: Detecting shadowing anomaly

the blank strip represents the corresponding area of the firewall accept rules, as shown in Figure 2. According to the definition of A_{sh}, shadowing anomaly detection is equivalent to locating the strip that filled with crossing lines. In this case, we first calculate the overlapping area of all the three upstream firewalls accepting rules, denoted as $FW_1 \wedge FW_2 \wedge FW_3$; Then we change the decision of rule in this area to *discard* and map it to the most downstream firewall FW_4; Finally, we obtain the shadowing anomaly A_{sh}.

Next, let us take FW_1 and FW_2 as examples to illustrate how to calculate firewall overlap. First, the decision of the strip area [1,9] representing FW_1 is changed to *discard* and mapped to the corresponding area [3,10] of FW_2, as shown in Figure 2. At this time, the uncovered area of FW_2 is [9,10]; Then we change the decision of these rules to *discard* and map them again to the original area [3,10] of FW_2, where the uncovered area [3,9] of FW_2 is the overlap of FW_1 and FW_2.

According to this method, the overlapping areas of FW_1,FW_2 and FW_3 can be calculated as [3,7] and [8,9]. Finally, the decision of these rules in the overlapping area is changed to *discard* and mapped to the most downstream firewall FW_4, where the uncovered area [7,8] of FW_4 is the shadowing anomaly A_{sh} we seek. This means that the most downstream firewall FW_4 allows packets in [7,8], which are blocked by upstream firewalls, that means a shadowing anomaly occurs in this area. In addition, based on this algorithm, we can obtain the shadowing anomaly between any two firewalls on the distributed firewall network path.

Take any network path in the distributed firewall network, assuming that there are n firewalls FW_1,FW_2,...,FW_n in the network path, where FW_1 is the most upstream firewall, FW_n is the most downstream firewall, and FW_i is the upstream firewall of $FW_j (i < j)$.

Our algorithm includes the following three steps: (1) map all upstream firewall rules of the most downstream firewall into a multidimensional matrix to form a set of independent unit spaces. (2) calculate the unit space overlap of all upstream firewalls. (3) generate the corresponding firewall rules from the overlapping unit space, change the rule decision to *discard* and map it to the multidimensional matrix corresponding to the most downstream firewall. In this case, the unit space area that uncovered by the most downstream firewall is called the Shadowing Anomaly A_{sh}.

1) Rule mapping and rule generating
   According to the rule mapping idea of FDM method, any rule with the form $F_1 \in D(F_1) \wedge \ldots \wedge F_k \in D(F_k) \rightarrow decision$ can be mapped to k-dimensional matrix M_k. In the mapping process, the area with *accept* decision is represented by independent unit spaces US (k-dimensional matrix unit). In order to generate the corresponding firewall rules from the unit spaces, all unit spaces are arranged in descending order according to their area size, and the corresponding firewall rules are respectively generated according to the sorted unit spaces.

2) Calculating the overlapping unit spaces
   For the two unit spaces US_i and US_j, we first generate the corresponding rule set R_i based on US_i, where the rule decision of R_i is *accept*; Then change their decision to *discard* and map it to US_j, at this time, the overlapping area of US_i and US_j can be covered by rule R_i, and then we take the uncovered area of US_j to generate the corresponding rule $R_{j-i \wedge j}$, change its decision to *discard* and map it to the original US_j again, here the unit space not covered in US_j is the overlapping part of US_i and US_j, which is recorded as $US_{i \wedge j}$.

Referring to the above descriptions (1) and (2), the specific process of detecting inter-firewall shadowing anomalies is described in Algorithm 1. The input of the main algorithm is n firewalls FW_1,...,FW_i,FW_j,...,FW_n in a network path, in which FW_i is the upstream firewall of $FW_j (i < j)$, the algorithm output is the Shadowing anomaly (A_{sh}).

4.3.2 Spuriousness Anomaly Detection

As shown in Figure 3, the blank strip represents the area corresponding to the firewall acceptance rule. Accordingly, spuriousness anomaly detection is equivalent to locating the shaded area filled with cross lines. We first calculate the overlapping area of all downstream firewalls, namely $FW_2 \wedge FW_3 \wedge FW_4$; Then we change the decision of the rule in this area to *discard* and map it to the most upstream firewall FW_1. The final area is the spuriousness anomaly A_{sp}.

Specifically, we first change the decision of the blank strip representing FW_2 to *discard* and map it to FW_3, as shown in Figure 3. At this time, the uncovered area of FW_3 is [8,10], we change the decision of this rule to *discard* and map it to the original area [3,10] of FW_3, here the uncovered area [3,8] of FW_3 is the overlapping part of FW_2

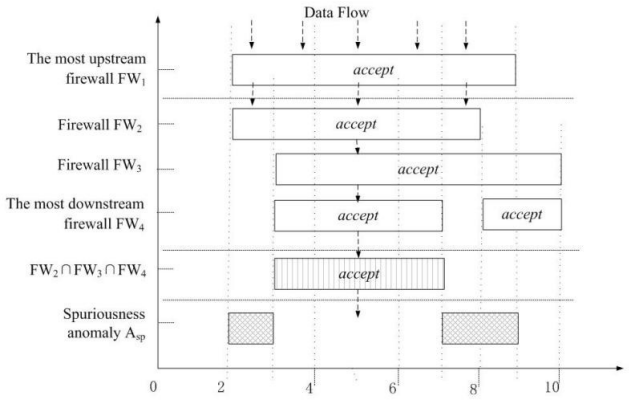
Algorithm 1 Inter-firewall Shadowing Anomaly Detection

1: Begin
2: map $FW_1,\ldots,FW_n$ into M_k to form a set of unit spaces $US_1,\ldots,US_n$.
3: **for** i:=1 to (n-2) **do**
4: $US_{i+1} \leftarrow$ ***Overlap***(US_i, US_{i+1}).
5: **end for**
6: generate rules R_{n-1} from US_{n-1}.
7: change the decision of R_{n-1} to *discard* and map them into US_n.
8: return the uncovered unit spaces in US_n, denoted as A_{sh}.
9: ***Overlap***(US_i, US_j)
10: generate rules R_i from US_i.
11: change the decision of R_i to *discard* and map them into US_j.
12: generate rules $R_{j-i\wedge j}$ from the uncovered unit space in US_j.
13: change the decision of $R_{j-i\wedge j}$ to *discard* and map them into US_j.
14: record the uncovered unit spaces in US_j, which is the overlap of US_i and US_j, denoted as $US_{i\wedge j}$.
15: return $US_{i\wedge j}$.
16: End

and FW_3; Then, we change the decision of rule in [3,8] to *discard* and map it to FW_4, and obtain the overlapping area [3,7] of FW_2, FW_3 and FW_4. Finally, the decision of the overlapping area is changed to *discard* and mapped to the most upstream firewall FW_1, where the uncovered areas [2,3],[7,9] of FW_1 is the desired spuriousness anomaly A_{sp}. This means that the most upstream firewall FW_1 permits packets in [2,3] and [7,9], which are blocked by the downstream firewall, thus spuriousness anomalies occur in this area. In addition, based on this algorithm, we can locate the spuriousness anomalies between any two firewalls on the network path of the distributed firewall environment.

The algorithm process includes the following three steps: (1) all downstream firewall rules of the most upstream firewall are mapped by FDM method respectively to obtain a series of independent unit spaces; (2) calculate the overlap of all unit spaces corresponding to the downstream firewall rules; (3) generate the corresponding firewall rules from the overlapping area, change the rule decision to *discard* and map them to the multidimensional matrix corresponding to the most upstream firewall, in which the uncovered unit space area of the most upstream firewall is the Spuriousness Anomaly A_{sp}. The input of the main algorithm is n firewalls $FW_1,\ldots,FW_i,FW_j,\ldots,FW_n$ in a network path, in which FW_i is the upstream firewall of $FW_j (i<j)$, the algorithm output is the Spuriousness anomaly (A_{sp}).

The execution process of function $Overlap(US_i, US_{i+1})$ is the same as that of Algorithm 1. According to the idea of our inter-firewall anomaly detection algorithm, the time



Note: A spuriousness anomaly occurs if an upstream firewall permits the network traffic denied by a downstream firewall.

Figure 3: Detecting Spuriousness anomaly

Algorithm 2 Inter-firewall Spuriousness Anomaly Detection

1: Begin
2: map $FW_1,\ldots,FW_n$ into M_k to form a set of unit spaces $US_1,\ldots,US_n$.
3: **for** i:=2 to (n-1) **do**
4: $US_{i+1} \leftarrow$ ***Overlap***(US_i, US_{i+1}).
5: **end for**
6: generate rules R_n from US_n.
7: change the decision of R_n to *discard* and map them into US_1.
8: return the uncovered unit spaces in US_1, denoted as A_{sp}.
9: End

complexity of the algorithm mainly depends on the number of firewalls on the network path of the distributed firewall and the execution efficiency of the FDM method. In [8], the time complexity of FDM method is analyzed in detail, which is $O(k^2n^2)$, where n and k are the number and dimension of firewall rules respectively. From the perspective of detection process, one of the main characteristics of this method is that all firewalls in the network path can be regarded as a whole and can comprehensively discover the anomalies caused by multiple rules, it is no longer limited to the traditional inter-firewall anomalies detection algorithm, which can only discover the anomalies between any two rules [1].

5 Performance Analysis and Simulation Results

5.1 Time Complexity Analysis

Suppose that there are t firewalls on the network path from the most upstream firewall FW_1 to the most downstream firewall FW_t. For simplicity, suppose the rules number of each firewall is n, then for Step (1) of Algo-

rithm 1, the time complexity of FDM mapping for t firewalls is $O(tk^2n^2)$. For Step (2), to calculate the overlap of all unit spaces of the upstream firewalls, the execution time of this step is mainly consumed in the circular execution function $Overlap(US_i, US_j)$, while there are two mapping operations during each round of function execution. The first mapping is to generate the rule set R_i from US_i, change its decision value to *discard*, and then overwrite and map it to US_j. Since the number of unit spaces obtained by FDM is generally not more than that of the original rules, the number of unit spaces contained in the t unit space sets is also not more than n. Considering the worst case, each unit space in US_i is included in that of US_j, when mapping each rule corresponding to US_i into a multidimensional matrix, an existing unit space in the multidimensional matrix will be divided into $2k$ sub-unit spaces. Therefore, the time required for the first mapping is:

$$\begin{aligned} & ck\sum_{i=1}^{n}(n-i+2ki) = ck\sum_{i=1}^{n}[n+(2k-1)i] \\ & = ck[n^2+(2k-1)\tfrac{n(n+1)}{2}] \\ & \le ckn^2+ck^2n(n+1) = ck(k+1)n^2+ck^2n \end{aligned} \tag{1}$$

At this time, a maximum of $2kn$ unit spaces in the multidimensional matrix are not covered. Let us continue to consider the second mapping operation, which includes the process of generating the corresponding rule set from the uncovered unit spaces $US_{j-i\wedge j}$ in US_j, changing the rule decision to *discard*, and mapping them into the original US_j. Different from the last mapping operation, there are $2kn$ rules that need to be mapped to the multidimensional space in turn. The time required is:

$$\begin{aligned} & ck\sum_{i=1}^{2kn}(n-i+2ki) = ck\sum_{i=1}^{2kn}[n+(2k-1)i] \\ & = ck[2kn^2+(2k-1)\tfrac{2kn(2kn+1)}{2}] \\ & \le 2ck^2n^2+ck^2n(n+1) = 4ck^3n^2+2ck^3n \end{aligned} \tag{2}$$

Considering the worst case, each unit space in US_i is included in US_j, so the number of unit spaces overlapped by US_i and US_j is exactly n, which means the overlapping $US_{i\wedge j}$ of US_i and US_j contains n unit spaces. The algorithm has t-2 cycles, so the time complexity of the algorithm is $O(tk^3n^2)$, where t is the total number of firewalls, k is the rule dimension and n is the number of rules.

For Step (3), a corresponding rule is generated from the overlapping area of $US_1,\ldots,US_{n-1}$, change the rules decision to *discard*, and map it to the multidimensional matrix space corresponding to the most downstream firewall. This step is equivalent to performing an FDM mapping operation. As mentioned earlier, in the worst case, the overlap of $US_1,\ldots,US_{n-1}$ has n unit spaces corresponding to n rules, so the time complexity of this mapping operation is $O(k^2n^2)$. Based on the above description, the total time of the algorithm is the sum of these three steps. Therefore, the worst-case time complexity of the algorithm is $O(T_{worst})$=$O(tk^3n^2)$.

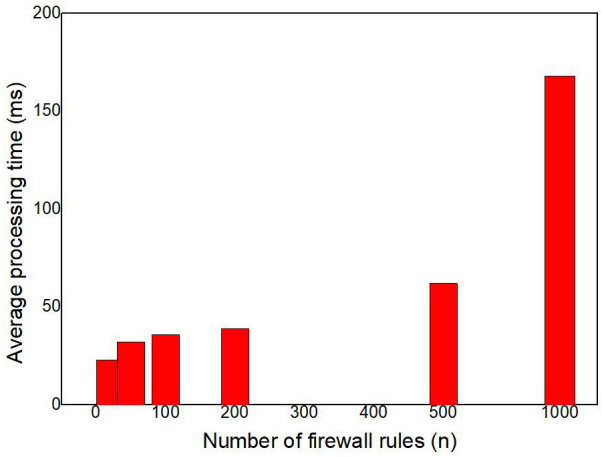


Figure 4: Intra-firewall: The average processing time for eliminating anomalies.

5.2 Experimental Results

In order to test the effectiveness of our method, we refer to the two virtual firewall policies given in [22], and use our algorithm to analyze the policy anomalies between the two firewalls. The specific configurations of the two firewalls are shown in Table 2 and Table 3. The upstream firewall FW_u contains ten rules and the downstream firewall FW_d contains seven rules.

Firstly, the intra-firewall anomaly analysis method is used to eliminate the intra-firewall anomalies in the upstream firewall FW_u and the downstream firewall FW_d respectively; Then, algorithm 1 and algorithm 2 are used to discover the shadowing anomaly and the spuriousness anomaly. The algorithms detects ten anomalies, including four shadowing anomalies and six spuriousness anomalies, which are consistent with the detection results in [22] and the actual situation. These six spuriousness anomalies are shown in Table 4.

Specifically, we first map FW_u and FW_d to M_k to form a set of unit spaces US_u and US_d respectively; Then we generate rules R_d from US_d, change the decision of R_d to *discard*, and map it to US_u; Finally, the uncovered unit spaces in the US_u are the spuriousness anomaly A_{sp} we desired.

For example, data packet "source IP ='B', destination IP ='M', source port ='*', destination port ='25', protocol ='TCP'" matches rule 3. Obviously, the packet is allowed to pass through the most upstream firewall FW_u, and blocked by the most downstream firewall FW_d. As mentioned earlier, if the upstream firewall permits network traffic denied by the downstream firewall, a spuriousness anomaly will occur. Therefore, rule 3 conforms to the definition of spuriousness anomaly.

In order to evaluate the efficiency of the proposed algorithm, we use *Classbench* [20] to generate six groups of firewall policies, each of which contains 20, 50, 100, 200, 500 and 1000 rules respectively. Each group con-

Table 2: The upstream firewall FW_u policy

	SourceIP	DestIP	SourcePort	DestPort	Prot	Action
1	A,B,C	H	*	80	TCP	*discard*
2	B,C	M,N	*	23,25	TCP	*discard*
3	*	M,N	*	*	TCP	*accept*
4	*	*	*	*	TCP	*discard*
5	C,D,E	H,K	*	53	UDP	*discard*
6	C,D,E	*	*	53	UDP	*accept*
7	*	*	*	*	UDP	*discard*

Table 3: The downstream firewall FW_d policy

	SourceIP	DestIP	SourcePort	DestPort	Prot	Action
1	D,E,F	O	*	80	TCP	*accept*
2	B	O	*	80	TCP	*accept*
3	B	M,N	*	25	TCP	*accept*
4	B	M,N	*	23	TCP	*accept*
5	E,F	*	*	139	UDP	*accept*
6	F	H	*	53	UDP	*accept*

Table 4: Spuriousness anomalies detection result

	SourceIP	DestIP	SourcePort	DestPort	Prot	Action
1	A,B,C	H	*	80	TCP	*discard*
2	A,B	M,N	*	23,25	TCP	*accept*
3	*	M,N	*	23,25	TCP	*discard*
4	A,C	O	*	*	TCP	*discard*
5	*	O	*	80	TCP	*accept*
6	*	*	*	*	TCP	*discard*
7	C,D,E	H	*	53	UDP	*discard*
8	*	H	*	53	UDP	*accept*
9	E,F	*	*	139	UDP	*accept*
10	*	*	*	*	UDP	*discard*

Table 5: The time(ms) of anomalies detection

Method	Anomalies	Link100	Link200	Link300	Link400	Link500
Method-W	A_{sh}	279.24	337.30	351.25	362.62	400.46
	A_{sp}	26.61	32.23	35.02	50.45	54.20
Method-C	A_{sh}	212.45	264.05	300.26	335.80	371.45
	A_{sp}	14.24	26.05	21.65	35.06	37.09

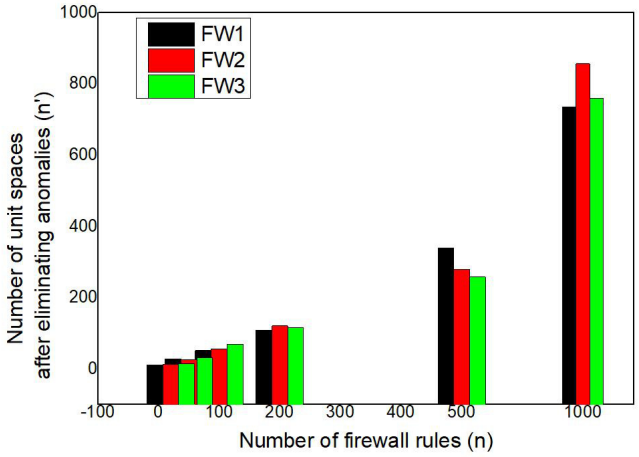


Figure 5: Intra-firewall: The number of rules after eliminating anomalies.

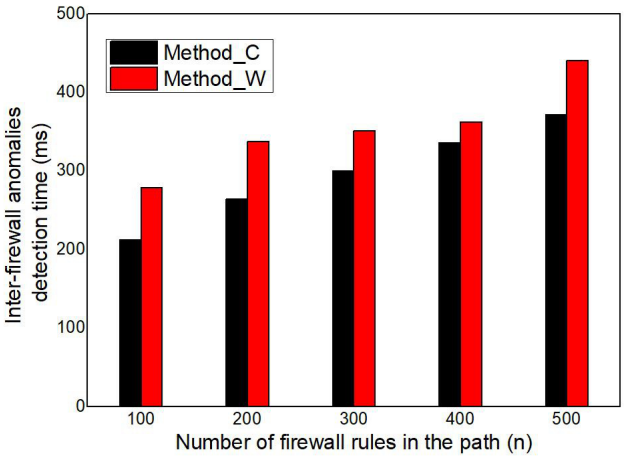


Figure 6: Processing time of detecting the inter-firewall anomalies

tains three firewall policies on average, named FW_1, FW_2 and FW_3. These algorithms were implemented in Java JDK 1.6, and we conducted our experiments on a desktop PC running Windows 7 with 4.0G memory and Intel(R) Core(TM) Processor of 2.60GHz.

We first eliminate the intra-firewall anomalies, the average running time of the program is shown in Figure 4. with the increase of the number of rules in firewall policy, the time required to eliminate policy anomalies increases accordingly. For example, when the number of firewall rules is 500, the system processing time is $65ms$, while when the number of firewall rules reaches 1000, the system only needs about $160ms$. It can be seen that the execution time of the system roughly conforms to the quadratic function curve, which also verifies the time complexity $O(k^2n^2)$ of FDM algorithm, and the execution efficiency of the method is high. Figure 5 shows the amount of unit space in each group of firewalls after eliminating policy anomalies. The result also verify that the number of unit spaces is less than the number of original rules.

In order to further evaluate the time performance of the inter-firewall spuriousness anomaly detection algorithm, we designed five different network paths to obtain the average processing time of the algorithm. For convenience, each path contains four firewalls with the same number of rules, expressed as FW_1, FW_2, FW_3, FW_4 from the most upstream firewall to the most downstream firewall. The number of firewall rules in these five network paths is 100, 200, 300, 400 and 500, respectively. Accordingly, we record them as path 100, path 200, path 300, path 400, and path 500. For example, path 100 means that there are four firewalls in the network path, and each firewall has 100 rules. The names of other paths follow the same principle. We execute algorithm 1 and algorithm 2 on the five paths respectively, the required running time of shadowing anomalies and spuriousness anomalies detected are shown in Table 5.

Figure 6 shows the time taken to detect the inter-firewall anomaly for each path using the *Method-C* herein and the *Method-W* described in [22], respectively. With the increase of the number of firewall rules in the path, the time required to detect firewall anomalies also increases. For a network path with four firewalls, especially when the number of rules in each firewall reaches 500, the method in this paper can detect the shadowing anomalies in only $370ms$. From the comparison results, it can be seen that our algorithm has higher execution efficiency.

6 Conclusions

This paper presents a method for detecting and eliminating the intra-firewall anomalies. This method can completely discover the anomalies while maintaining the consistency, compactness and completeness of the original firewall rules. Then the definition and classification of inter-firewall anomalies are discussed, and an approach of inter-firewall anomalies detection based on rule space comparison is proposed. Theoretical analysis and simulation results show that this method can detect shadowing anomalies and spuriousness anomalies accurately and efficiently.

Acknowledgments

This study was supported by Research Foundation of the Education Department of Hunan Province (No. 21C1589), the Doctoral Scientific Research Project of Changsha Social Work College(2020JB32), and the National Natural Science Foundation of China (61877059). The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] E. S. Al-Shaer and H. H. Hamed, "Discovery of policy anomalies in distributed firewalls," in *Proceedings of The IEEE INFOCOM (INFOCOM'04)*, pp. 2605–2616, Hong Kong, China, March 2004.

[2] J. G. Alfaro, F. Cuppens, and N.Cupperns-Boulahia, "Analysis of policy anomalies on distributed network security setups," in *Proceedings of The European Symposium on Research in Computer Security (ESORICS'06)*, pp. 496–511, Hamburg, Germany, Sept. 2006.

[3] F. Baboescu, S. Singh, and G. Varghese, "Packet classification for core routers: is there an alternative to cams?," in *Proceedings of The IEEE INFOCOM (INFOCOM'03)*, pp. 53–63, San Francisco, USA, March 2003.

[4] D. Bringhenti, G. Marchetto, and R. Sisto, "Automated optimal firewall orchestration and configuration in virtualized networks," in *Proceedings of The IEEE/IFIP Network Operations and Management Symposium (NOMS'20)*, pp. 1–7, Budapest, Hungary, April 2020.

[5] C. S. Chao, "A flexible and feasible anomaly diagnosis system for internet firewall rules," in *Symposium of The Asia-Pacific Network Operations and Management (APNOMS'11)*, pp. 1–8, Taipei, Taiwan, Sept. 2011.

[6] F. Chen, A. X. Liu, and J. H. Hwang, "First step towards automatic correction of firewall policy faults," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 7, no. 2, pp. 1–15, 2012.

[7] Y. X. Cheng, H. J. Yao, Y. Wang, Y. Xiang, and H. P. Li, "Protecting vnf services with smart online behavior anomaly detection method," *Future Generation Computer Systems*, vol. 95, pp. 265–276, 2019.

[8] Y. Z. Cheng, W. P. Wang, G. Y. Min, and J. X. Wang, "A new approach to designing firewall based on multidimensional matrix," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 12, pp. 3075–3088, 2015.

[9] Y. Z. Cheng, W. P. Wang, J. X. Wang, and H. D. Wang, "Fpc: a new approach to firewall policies compression," *Tsinghua Science and Technology*, vol. 24, no. 1, pp. 65–76, 2019.

[10] M. G. Gouda and A. X. Liu, "Structured firewall design," *Computer Networks*, vol. 51, no. 4, pp. 1106–1120, 2007.

[11] H. H. Hamed and E. Al-Shaer, "Taxonomy of conflicts in network security policies," *IEEE Communications Magazine*, vol. 44, no. 3, pp. 134–141, 2006.

[12] H. X. Hu, G. Ahn, and K. Kulkarni, "Detecting and resolving firewall policy anomalies," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 3, pp. 318–331, 2012.

[13] J. H. Hwang, T. Xie, F. Chen, and A. X. Liu, "Systematic structural testing of firewall policies," *IEEE Transactions on Network and Service Management*, vol. 9, no. 1, pp. 1–11, 2012.

[14] M. S. Hwang and W. P. Yang, "Controlling access in large partially-ordered hierarchies using cryptographic key," *The Journal of Systems and Software*, vol. 67, no. 2, pp. 99–107, 2003.

[15] T. Kim, T. Kwon, and J. Lee, "F/wvis: hierarchical visual approach for effective optimization of firewall policy," *IEEE Access*, vol. 9, pp. 105989 – 106004, 2021.

[16] L. H. Liu, Z. J. Cao, and M. Chong, "A note on one outsourcing scheme for big data access control in cloud," *International Journal of Electronics and Information Engineering*, vol. 9, no. 1, pp. 29–35, 2018.

[17] B. E. Logan and G. G. Xie, "Automating distributed firewalls: a case for software defined tactical networks," in *Proceedings of The IEEE Military Communications Conference (MILCOM'19)*, pp. 1–6, VA, USA, Nov. 2019.

[18] L. Maccari and R. L. Cigno, "Privacy in the pervasive era: a distributed firewall approach," in *Proceedings of The 9th Annual Conference on Wireless On-Demand Network Systems and Services (WONS'12)*, pp. 23–26, Courmayeur, Italy, Jan. 2012.

[19] S. R. Pedditi, D. Zhang, and C. Wang, "Fiep: an initial design of a firewall information exchange protocol," in *Proceedings of The 14th International Conference on Information Reuse and Integration (IRI'13)*, pp. 1–5, San Francisco, USA, Aug. 2013.

[20] D. E. Taylor and J. S. Turner, "Classbench: a packet classification benchmark," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 499–511, 2007.

[21] C. Togay, A. Kasif, C. Catal, and B. Tekinerdogan, "A firewall policy anomaly detection framework for reliable network security," *IEEE Transactions on Reliability*, vol. 99, pp. 1–9, 2021.

[22] W. P. Wang, W. H. Chen, W. P. Zhu, H. P. Chen, and J. Yang, "Analysis of distributed firewall policy configuration mistakes and their detection," *Journal of University of Chinese Academy of Science(in Chinese)*, vol. 24, no. 2, pp. 257–265, 2007.

[23] S. Yingchareonthawornchai, J. Daly, A. X. Liu, and E. Torng, "A sorted-partitioning approach to fast and scalable dynamic packet classification," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1907–1920, 2018.

[24] L. H. Yuan, J. N. Mai, Z. D. Su, H. Chen, C. N. Chuah, and P. Mohapatra, "Fireman: a toolkit for firewall modeling and analysis," in *Proceedings of The IEEE Symposium on Security and Privacy (SSP'06)*, pp. 199–213, California, USA, Jan. 2006.

Biography

Yu-zhu Cheng received the B.S. degree from Hunan university of science and technology in 2002 and the M.S. degree in software engineering from Hunan University in 2005, and the Ph.D. degree in computer science and technology from Central South University

in 2018. Currently, he is an associate professor at Changsha Social work College. His current research interests include data privacy and information security. He has published more than 30 papers in referred journals and conference proceedings, he is a member of IEEE.

Qiu-ying Shi received the B.S. degree from Hunan Normal University and the M.S. degree in computer science and technology from Central South University. Her research interests include cyber security and pattern recognition.