Towards a Usable Anomaly Diagnosis System among Internet Firewalls' Rules

Chi-Shih Chao[1], Stephen J. H. Yang[2]

[1] Communications Eng. Department, Feng Chia University, Taiwan
[2] Computer Sciences and Information Eng. Department, National Central University, Taiwan
cschao@fcu.edu.tw, Stephen.Yang.Ac@gmail.com

Abstract

While configuring firewalls, firewall rule editing, ordering, and distribution must be done with extreme caution on each of cooperative firewalls. However, network operators are prone to incorrectly configuring firewalls because commonly there are hundreds of thousands of filtering rules (i.e., rules in the Access Control List file; or ACL for short) which could be set up in a firewall, not mention these rules among firewalls can affect mutually. To complete the crucial but laboring inspection of rule configuration on firewalls effectively and efficiently, this paper describes two of our developed diagnosis mechanisms which can speedily discover rule anomalies within/among firewalls with two innovative data structures – Rule Anomaly Relationship tree (RAR tree) and Adaptive RAR tree (ARAR tree). With the assistance of these data structures and associated algorithms, two of our developed mechanisms show significant improvements on system performance and scalability in rule anomaly diagnosis for Internet firewalls.

Keywords: Defense in depth, Firewall rule anomalies, RAR tree, ARAR tree, Diagnosis reuse

1 Introduction

Firewalls have become the emblem for modern network security where they function to avoid unauthorized or illicit sessions established to the devices inside the network areas it protects. In most of cases, a swarm of firewalls would be deployed in the proper positions of the managed network for cooperative, integrated, and in-depth network security protection [1]. However, in a large and complex network equipped with a stack of firewalls, it is very possible for a network manager to make mistakes while setting the firewall rules (i.e., ACL rules) since maintaining the security consistency between firewalls' rule configurations and the demands of network security policies is always time-consuming, laboring, and error-prone [2], due to the lower-level programming language properties of those filtering rules. Sometimes, the matter can go evern worse where there are a couple of network administrators with different level of professional knowledge who are assigned to do this job collectively, setting up the filtering rule on different timings. This explains why researchers [3] like to compare the firewall configuring task to programming a distributed system in assembly language.

Such kind of the security inconsistencies typically can be revealed by either the occurrence of anomalies between the firewall rules or demand-mismatching of network security policies [4]. Among the literature in this field, E. Al-Shaer and H. Hamed first define a rule anomaly as a duplicate or multiple rule-matching for a packet in a rule set. Based on the concept, they formally define several different intra-/inter-ACL anomalies among the firewall rules. Nevertheless, since a Finite-State-Machine (or FSM)-based comparison between each pair of filtering rules should be conducted for rule anomaly check-ups, their rule anomaly inspection algorithm will meet an inefficiency as the number of rules or firewalls grows [5].

To lower the comparisons between firewall rules needed in [4], Y. Yin et al. [6] segment the IP address space, which is formed by the managed source and destination networks, into blocks where each block is precisely split by the IP addresses in the conditional field of each firewall rule. Utilizing these varying-sized blocks, a SIERRA tree is built and two conflict rules would be hanged on the same branch of the tree [7]. Network managers only needs to do the anomaly inspections or check-ups on rules in the same spatial block(s) (or on the same branches in the SIERRA tree), as opposing to wasting enormous time to conduct a comprehensive pair-wise rule comparisons. Yet, this approach would lead to a fatal drawback in a networking environment with frequent rule updates. Besides, a clean-slate reconstruction of the SIERRA tree is very possibly unavoidable if a simple rule deletion or insertion is administored [8]. It is because space blocks are precisely sliced according to the IP addresses of each rule. So, once one rule changes, the

*Corresponding Author: Chi-Shih Chao; E-mail: cschao@fcu.edu.tw
DOI: 10.3966/160792642019052003012

whole spatial rule relationship would change, and the corresponding data structures could be reconstructed. This drawback also reveals that the local diagnosis results, i.e., the intra-ACL diagnosis results, can hardly be re-used for the diagnosis of inter-ACL rule anomalies. By the same token, modification or reconfiguration of firewall rules for new demands of network security could fail their system to go live in time for varying threats.

In our work, our goal is to build a feasible diagnosis system to ease the discovery of the anomalies between firewalls' rules where the developed system should take efficiency, effectiveness and scalability into account. To describe how we achieve the goals in our work, in the rest of this paper, we organize Session 2 to spell out how a Rule Anomaly Relation tree (RAR tree) is created on the basis of these collected firewall rules, and how it can be used in our system to facilitate the diagnosis of intra/inter-ACL rule anomalies. In Session 3, an improved version of RAR tree-based diagnosis system with new data structure – ARAR tree is shown. The corresponding diagnosis mechanism for intra-ACL and inter-ACL rule anomalies are rendered also. As a demonstration, Section 4 presents the system implementation and performance evaluations for our develped diagnosis mechanisms, and Section 5 concludes this paper and shows some directions of our system development in the upcoming future.

2 Anomaly Diagnosis with RAR Tree

For the anomalies between firewall rules, they are completely defined by E. Al-Shaer et al and classified broadly into two types: anomalies within one single ACL (or referred to as intra-ACL rule anomalies) and anomalies among different ACLs (or called inter-ACL rule anomalies) [4]. In this section, we will introduce our RAR tree-based diagnosis approach and show how it aids us in achieving the system development goal: Effeectiveness, effieincy, and scalability.

2.1 Discovery of Intra-ACL Rule Anomalies

Figure 1 contains the example network used throughout this section to describe how our RAR tree is built and facilitates the diagnosis of intra-ACL rule anomalies. For brevity, Figure 2 only shows those rules in firewall H which manage the flow(s) from IP address(es) in domain D2 (140.134.30.*) to IP address(es) in Domain D7 (152.127.10.*) with service port numbered 80. An ACL consists of a number of filtering rules in the form of (<order>, <protocol>, <source_IP>, <source_port>, <destination_IP>, <destination_port>, <action>) to do packet filtering on a specific interface of a firewall with direction indications (for outbound or inbound traffic filtering). To avoid the typical time-consuming pair-wise rule comparisons for anomalies check-ups [4-5], a 2-dimensional address space matrix is designed as a structural basis of our RAR tree to root out those unnecessary comparisons in which there is no intersection (or overlap) between the IP address spaces of two rules. To do so, in our system, the IP address ranges of the source network domain and destination network domain are used as two axes to construct a rectangle plane which is further divided into a matrix consisting of blocks in size of A*A. Later, with the fields of <source_IP> and <destination_IP>, the IP address space of each ACL rule can be represented as a smaller rectangle and drawn on the proper place of this matrix. For example, as shown in Figure 3, the 2-dimensional address space matrix constructed by domains D2 and D7 can be divided into 64 blocks in size of 32*32 IP addresses, in which the IP address space of each rule in Figure 2 is depicted as a smaller rectangle (or even a point) and put on its own appropriate location on the matrix.

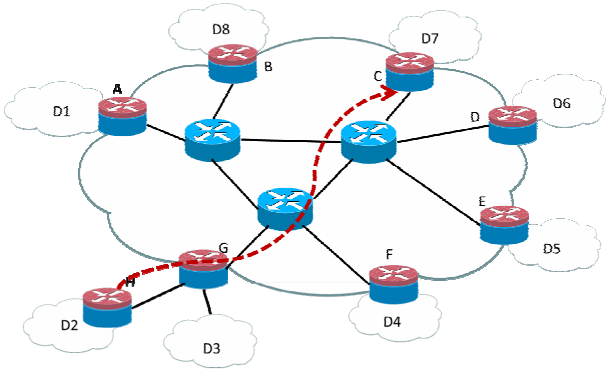

Figure 1. Example network for our rule anomaly diagnosis

Order	Source IP Address	Destination IP Address	Action
1.	140.134.30.48~140.134.30.80	152.127.10.48~152.127.10.80	accept
5.	140.134.30.32~140.134.30.95	152.127.10.32~152.127.10.127	deny
7.	140.134.30.128~140.134.30.158	152.127.10.128~152.127.10.158	accept
21.	140.134.30.96~140.134.30.223	152.127.10.192~152.127.10.223	accept
22.	140.134.30.220	152.127.10.220	accept
39.	140.134.30.192~140.134.30.223	152.127.10.192~152.127.10.253	deny
40.	140.134.30.192~140.134.30.255	152.127.10.1~152.127.10.253	accept
51.	140.134.30.70	152.127.10.35	deny
99.	*.*.*.*	*.*.*.*	deny
			80 port

Figure 2. Portion of ACL rules for HTTP traffic in firewall H from D2 to D7

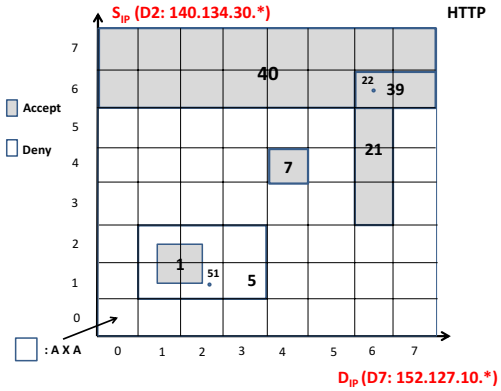

Figure 3. Two-dimensional address space matrix

After that, the address space of a rule can be recorded in our RAR tree in the form of □—○—△, where □ contains the values of the conditional fields of the rule, ○ is used to indicate the matrix block(s) spanned by the address space of the rule, △ shows the label (the order number) of the rule. As an example, in Figure 3, the address space of rule 1 in Firewall H spans four matrix blocks which are numbered block(1-1), block(1-2), block(2-1), and block(2-2). In this case, our RAR tree will use four branches to record them for the rule, i.e., the portion enclosed by the left rectangle in Figure 4. By handling each rule in Figure 2 in this fashion, the RAR tree depicting the structural configuration of Figure 3 can be built as Figure 4. From Figure 4, it can be found that there are six branches containing more than one △ leaves, which indicates only the IP address spaces of those rules in these branches could have the chance to intersect (or overlap) with each other and hence incur intra-ACL rule anomalies. So, we only have to do the pair-wise rule comparisons for anomaly checking on the rules at the same branch within these six branches; i.e., in our approach, we need to conduct the FSM-based pair-wise rule comparisons on H.5 and H.1, H.51 and H.1, H. 51 and H.5, H.22 and H.21, H.39 and H.21, H.40 and H.21, H.39 and H.22, H.40 and H.39, and H.40 and H.22 where *x.y* indicates the *y*th rule in firewall *x*.

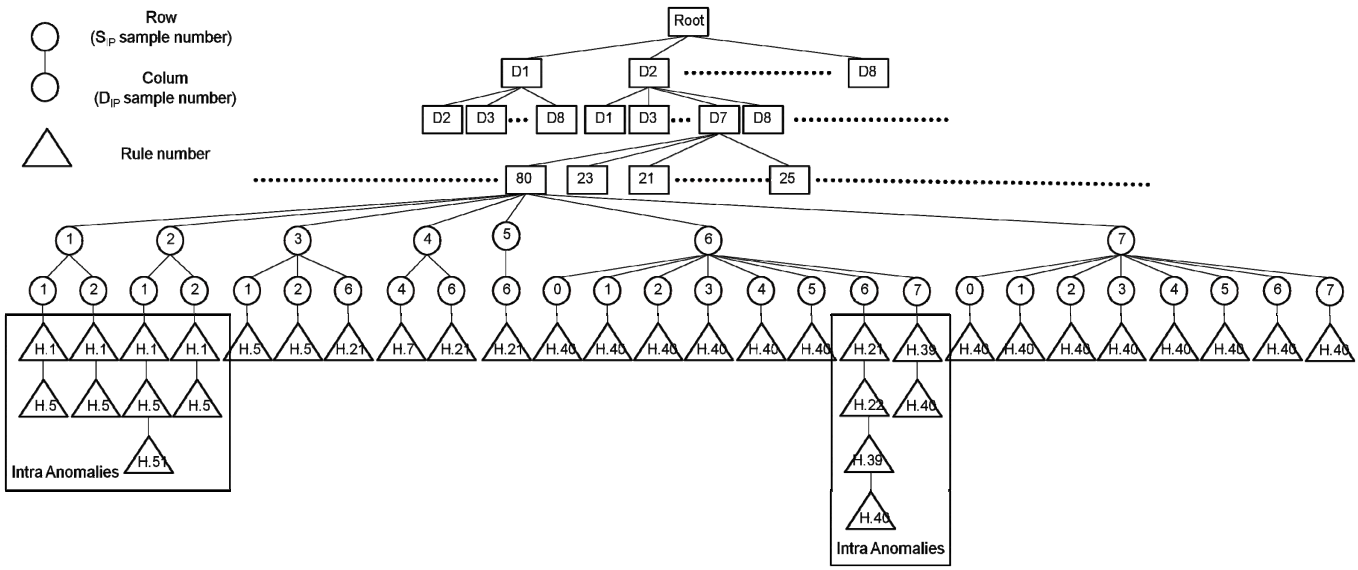

Figure 4. The RAR tree of Figure 3

It shows there are only 9 comparisons in total that should be made with our method. Comparing to [8], if the RAR tree is not employed, then $C_2^8 = 28$ rule pair-wise comparisons are required for anomaly inspection.

2.2 Discovery of Inter-ACL Rule Anomalies

To isolate the inter-ACL (or even inter-firewall) rule anomalies, in our system, it can easily be achieved by simply re-using the RAR trees built for the diagnosis of intra-ACL (or intra-firewall) rule anomalies. Following the same example in Figure 1, let us assume that Figure 5 only shows those rules in firewall G which manage the same flows from IP addresses in domain D2 (140.134.30.*) to IP addresses in Domain D7 (152.127.10.*) with service port numbered 80. If we want to do the diagnosis of inter-firewall rule anomalies between firewalls H and G, the way described in the following paragraph is utilized in our system to achieve the system scalability and flexibility.

As network managers often administor, we can first do the intra-ACL anomaly diagnosis for rules inside firewall H and firewall G individually. As described in the previous subsection, this will accompany the construction of two RAR trees separately for the

Order	Source IP Address	Destination IP Address	Action
1.	140.134.30.96~140.134.30.159	152.127.10.120~152.127.10.140	accept
28.	140.134.30.128~140.134.30.159	152.127.10.141~152.127.10.170	deny
70.	140.134.30.2	152.127.10.30	deny
72.	140.134.30.96~140.134.30.223	152.127.10.192~152.127.10.223	accept
79.	140.134.30.230	152.127.10.240~152.127.10.255	accept
80.	*.*.*.*	*.*.*.*	deny
			80 port

Figure 5. Portion of ACL rules for HTTP traffic in firewall G from D2 to D7

diagnosis of intra-ACL rule anomalies within firewall H and G. Later, to obtain the diagnosis of inter-firewall rule anomalies between these two firewalls, a tree integration can be made by simply collecting the leave △ nodes belonging to the same branch of the two individual RAR trees and putting them together under the same branch of a new RAR tree for inter-ACL rule anomaly diagnosis. Figure 6 shows the results of tree integration where the leave nodes of Firewall H are represented by white △ and the leave nodes of Firewall G are represented by gray △. Following the same logic described in Section 2.1 for the diagnosis of intra-ACL rule anomalies, the pair-wise comparisons for the diagnosis of inter-ACL rule anomalies would merely be conducted for those rules which are under the same

branch of the integrated RAR tree for inter-ACL rule anomaly diagnosis since only the IP address spaces of those rules have the possibilities to intersect in the address matrix block indicated by the branch of the integrated RAR tree.

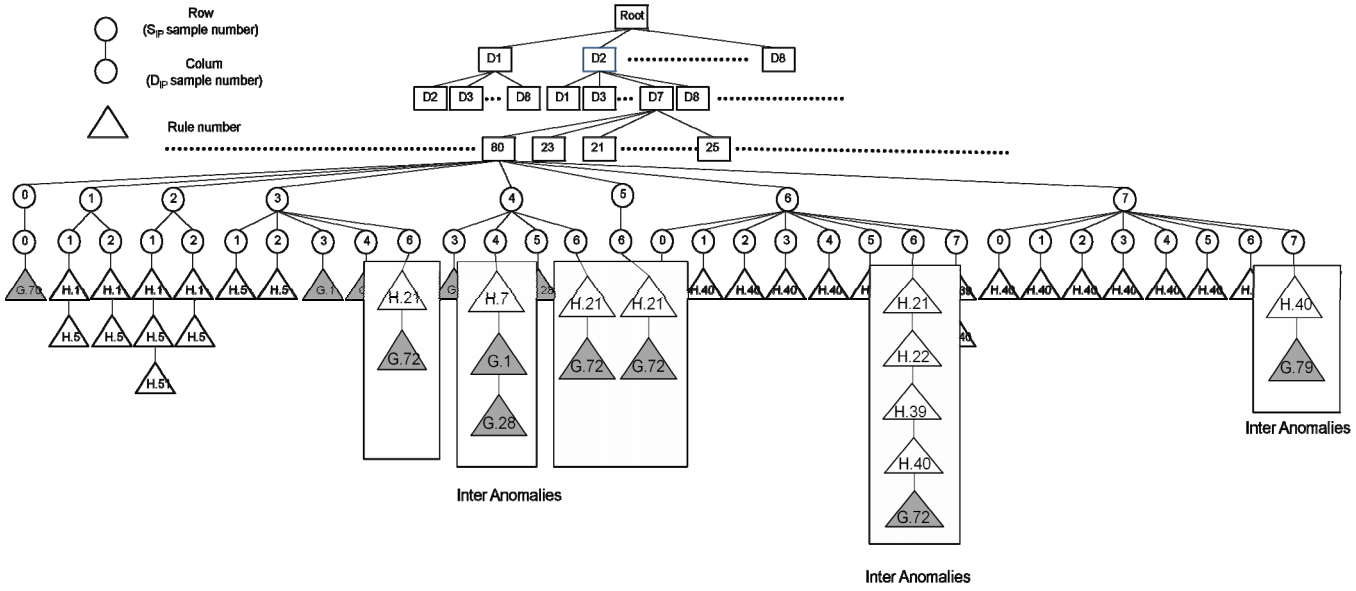

Figure 6. The RAR tree for the Inter-ACL rule anomaly diagnosis between firewalls H and G

As a result, two dominating advantages can be obtained by the introduction of our RAR tree:

(1) Unlike the existing approaches [4, 6], in our system, the local diagnosis results can be fully and easily reused where the RAR trees built for intra-ACL rule anomaly diagnosis can be easily integrated for the use of the inter-ACL rule anomaly diagnosis. In the case of Figure 6, our system only has to check those rules which are enclosed by five rectangles for inter-ACL rule anomalies. That is, only 7 FSM-based pair-wise comparisons for inter-ACL rule anomalies (i.e., H.21 and G.72, H.7 and G.1, H.7 and G.28, H.22 and G.72, H.39 and G.72, H.40 and G.72, and H.40 and G.79) should be made by using our RAR tree-based approach. Comparing with the traditional pair-wise-based solution in [4] which needs 9*5=45 (number of rules in Firewall H * number of rules in Firewall G for traffic from D2 to D7) inter-ACL rule anomaly comparisons, our system makes a huge saving of about 84% time-consuming comparisons for inter-ACL rule anomaly check-ups in this case.

(2) By simple integration of RAR trees for intra-ACL rule anomaly diagnosis, it can be seen that our system can easily deal with the diagnosis of the inter-ACL rule anomalies among a large number of firewalls in an enterprise-level network. It represents our RAR tree-based diagnosis has superior scalability of being

up against for network expansion. In contrast, it is hard for those approaches [4, 6] which heavily depend on packet classification-like approach or rule pair-wise comparison to attain to this goal since the data structures and the diagnosis results of the intra-ACL rule anomaly diagnosis can not be integrated or re-used to facilitate the inter-ACL rule anomaly diagnosis. In addition, another thing which should be noticed is that it is quite easy to provoke a substantial change on the associated data structure(s) of those approaches when a rule update (e.g., a rule insertion or deletion) is needed [8]. Such "clean-slate" approaches would do much more efforts on the re-building of data structures for the inter-ACL rule anomaly diagnosis. As a result, low system expansibility and scalability is incurred.

3 Anomaly Diagnosis with ARAR Tree

In spite of great improvement on diagnosis performance, the major problem with our RAR tree-based method is that, for users, it is quite difficult to choose an appropriate value of A (i.e., the size of fundamental blocks) beforehand to minimumize the pair-waise comparisons needed in different conditions [9]. This would dramatically deduce the usability of the system so that improvements are imperative. In addition, among the anomalies proposed in [4], it can be found that those anomalies in which the filtering spaces of two conflicting rules (i.e., two rules with opposite actions; one is 'accept' and the other is 'deny') have intersections or overlaps can have chance to creat security inconsistency or flaws in reality. The two findings mentioned above motivate the creation of the new version of our system – the ARAR (Adaptive RAR) tree-based diagnosis mechanism.

3.1 Discovery of Intra-ACL Rule Anomalies

To have an insight of our ARAR tree-based diagnosis mechanism, another ACL rule set shown in Figure 7 is used where those filtering rules for HTTP service are configured in firewall H in Figure 1, for the

routing path from network domain D2 to domain D7 (the dotted line in Figure 1). In this version of our system, the IP address ranges of the source network domain and destination network domain of a designated routing path are employed again as two axes to form a rectangle traffic plane; later, with the fields of <source_IP> and <destination_IP>, the IP address space of each ACL filtering rule can be depicted as a smaller rectangle and put on the proper location of this traffic plane (shown Figure 8).

Firewall H, Port 80				
Name	Order	Source IP Address	Destination IP Address	Action
R1	1	192.168.0.128~192.168.0.223	192.168.1.64~192.168.1.95	accept
R2	5	192.168.0.160~192.168.0.223	192.168.1.0~192.168.1.95	deny
R3	7	192.168.0.128~192.168.0.159	192.168.1.0~192.168.1.95	deny
R4	21	192.168.0.0~192.168.0.63	192.168.1.0~192.168.1.223	accept
R5	22	192.168.0.32~192.168.0.63	192.168.1.128~192.168.1.223	deny
R6	39	192.168.0.32~192.168.0.127	192.168.1.128~192.168.1.159	deny
R7	40	192.168.0.96~192.168.0.127	192.168.1.128~192.168.1.223	accept

Figure 7. Another rule set for HTTP traffic from D2 to D7 in firewall G

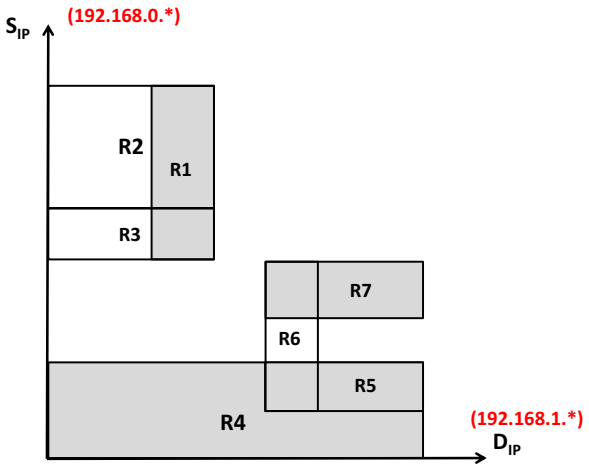

Figure 8. Two-dimensional traffic filtering plane for the rule set in Figure 7

Referring to the coding-tree data structures widely used in image/video compression [10], the traffic plane will be split recursively and reverse-exponentially (from Figure 9 to Figure 11) until a split block finds (1) there is no rule filtering space within it (e.g., Quadrant labeled by 11 in Figure 9), (2) there is only one rule filtering space within it (e.g., Quadrant labeled by 00 in Figure 9), or (3) there are more than two rule filtering spaces within it and the split block is exactly the same as those rule filtering spaces (e.g., Quadrant labeled by a binary string 110011 and highlighted by a circle in Figure 11; the split block is fully equal to the filtering spaces of R4 and R5 at that place), rather than splitting the traffic plane into a matrix which consists of fixed-sized smaller blocks as our RAR tree-based mechanism does.

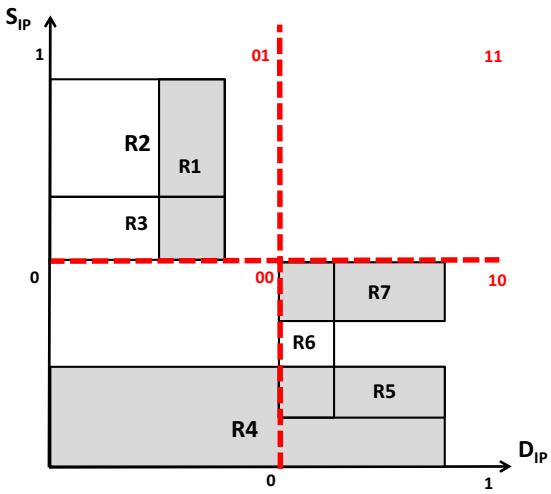

Figure 9. Traffic plane after 1st splitting

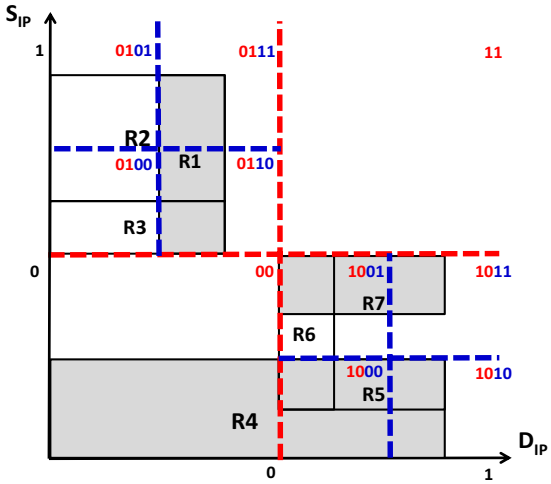

Figure 10. Traffic plane after 2nd splitting

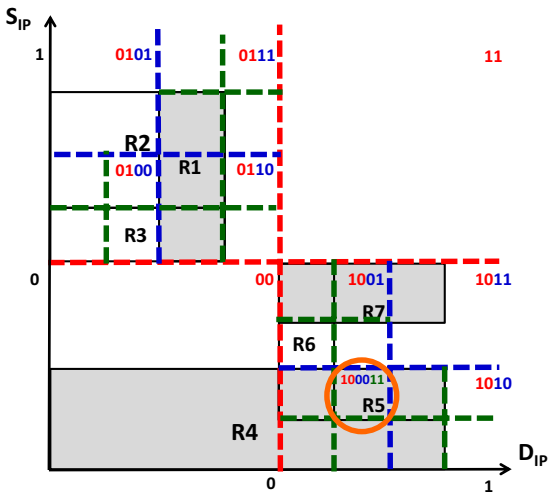

Figure 11. Traffic plane after 3rd splitting

After that, the address space of a filtering rule can be recorded in our ARAR tree in the form of □—□—○—...—○—△, where □ contains the IP address ranges of the source network domain and destination network domain of a designated routing path, ○ is used to indicate the split block(s) spanned by the address space of the rule, △ shows the label (or the order) of the rule. By dealing with each rule in this fashion, the ARAR tree depicting the structural configuration of Figure 11 can be created as Figure 12. From Figure 12, we can find that there are nine branches containing more than one △ leaves and highlighted by red rectangles, which indicates the IP address spaces of those rules under these branches intersect with one another and hence incur intra-ACL rule anomalies. Later, we simply have to do the checking for the type of rule anomlies on those rules, instead of the time-consuming pair-wise rule comparisons for anomaly check-ups which are needed in our RAR tree-based system and the system developed in [1].

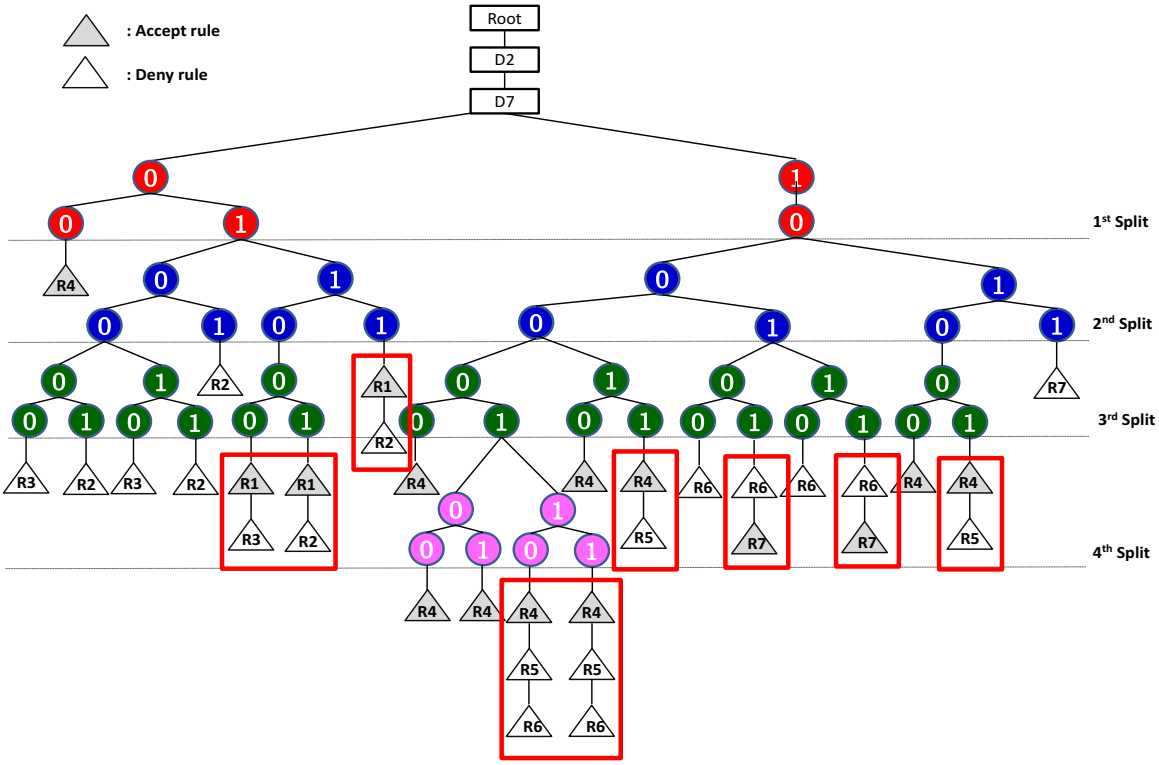

Figure 12. The corresponding ARAR tree of Figure 11

The key of success to our ARAR tree–based diagnosis mechanism is the use of recursive splitting on the traffic plane. The following highlights and summerizes how the two-dimenional traffic plance is split:

Step 1: Get the largest S_{IP} and D_{IP} from the ACL rule set for a specific serivce (e.g., port 80). Input them separately to *generateRSV*() (shown in Figure 13) to acquire the initial regional horizontal and vertical Regional Split Vales (RSVs).

```
Algorithm generateRSV (max_IP)
==========================================
Input: max_IP // max SIP or DIP
Output: initRSV // initial RSV

Begin
   While (max_IP & (max_IP - 1))
      max_IP & (max_IP - 1);
return max_IP
End
```

Figure 13. Pseudo Code of *generateRSV*()

Step 2: Split the 2-dimensional traffic plance by utilizing the two RSVs and build the corresponding ARAR tree.

Step 3: Check if there is a branch where there are more than one leave node under the branch with different filtering area sizes. If the answer is yes, go to Step 4; otherwise, go to Step 5.

Step 4: Halve the two RSVs, respectively, and then use them to adjust the two new RSVs for splitting the region indicated by the branch (like Figure 10 and Figure 11). Go to Step 2.

Step 5: Splitting stops. Check which types of anomalies occur. If two leaves of rules under a branch with different actions (i.e., one is accept and the other is deny), it is correlated/shadowing anomaly. If two leaves of rules under a branch with the same actions, redundant anomaly occurs in that region indicated by the branch.

Let's give a simple example to show how

generateRSV() can figure out the initial RSVs. Recalling from the example in Figure 7, the max S_{IP} is 192.168.0.223 and max D_{IP} is 192.168.1.223. Input the the two values into the *generateRSV*(), we get two RSVs (128.0.0.0, 128.0.0.0). It means that, at the first round, the traffic plane would be segmented into four quadrants, just like Figure 9.

3.2 Discovery of Inter-ACL Rule Anomalies

To isolate the inter-ACL (or, in general, inter-firewall) rule anomalies, in our approach, it can easily be done by simply re-using the ARAR trees built for the diagnosis of intra-ACL (or intra-firewall) rule anomalies. We can first do the intra-ACL anomaly diagnosis for rules inside two designated ACLs/firewalls individually, which can bring to the construction of two ARAR trees separately for the diagnosis of intra-ACL rule anomalies. Later, to obtain the diagnosis of inter-ACL (or inter-firewall) rule anomalies between these two firewalls, tree integration can be accomplished by adjusting the trees, collecting the leave △ nodes belonging to the same branch of the two individual ARAR trees, and putting them together under the same branch of a new ARAR tree for inter-ACL rule anomaly diagnosis. Later, following the same logic in our diagnosis for intra-ACL rule anomalies, the checking on anomaly types for the diagnosis of inter-ACL rule anomalies would be conducted only for those rules which are under the same branch of the integrated ARAR tree for inter-ACL rule anomaly diagnosis.

For a clear understanding of the integration process of two individual ARAR tree, Figure 14 and Figure 15 are created as the traffic places configured in firewall C and Firewall H, respectively, for the the HTTP traffic from network domain D2 to D7 in Fig. 1, where Figure 16 shows the integrated ARAR tree for the diagnosis of inter-ACL rule anomaly diagnosis between these two firewalls.

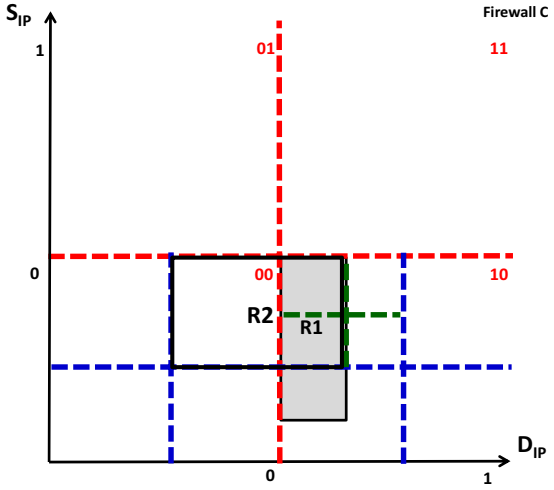

Figure 14. Traffic plane for HTTP traffic from D2 to D7 in firewall C

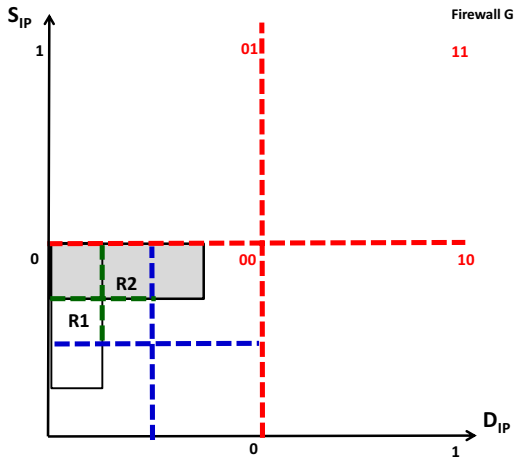

Figure 15. Traffic plane for HTTP traffic from D2 to D7 in firewall H

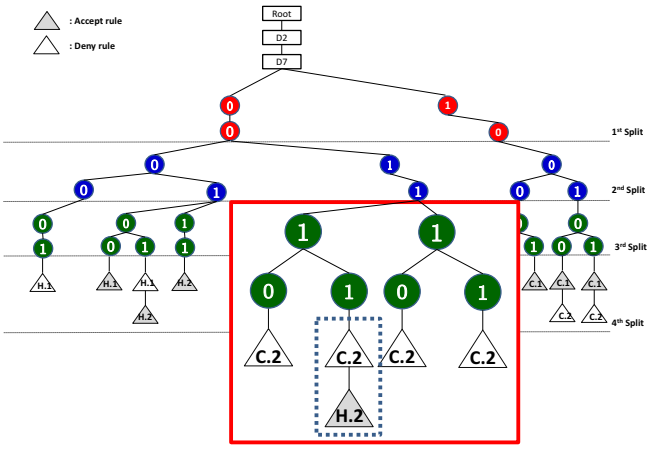

Figure 16. Integrated ARAR tree for Figure 14 and Figure 15

Algorithm in Figure 17 shows the integration process of two individual ARAR trees for Inter-ACL rule anomaly diagnosis. As an example, the area in Figure 16 highlighted by a rectangle complies with the case 4 of the algorithm in Figure 17. As for the time complexity of the algorithm, it is highly related to the height of the ARAR tree, which can be represented as $O(\log_2(\max(D_{IP}, S_{IP})))$.

Algorithm ARAR_Tree_Integration (T_1, T_2)

Input: Two ARAR trees, pointed by T_1, T_2
Output: A new ARAR tree, pointed by T_{new}

Begin
  if node(T_1) is null and node(T_2) is null
    then stop;
else if node(T_1) is not null but node(T_2) is null
    then node(T_{new}) is node(T_1); // case 1
  else if node(T_1) is null but node(T_2) is not null
    then node(T_{new}) is node(T_2); // case 2
  else if node(T_1) is leaf and node(T_2) is leaf

then node(T_{new}) is node(T_1) + node(T_2); // case 3
else if node(T_1) is leaf but node(T_2) is not leaf
then create 4 branches for node(T_1): // case 4
node(T_1)-0-0-leaf, node(T_1)-0-0-leaf,
node(T_1)-0-0-leaf, and node(T_1)-0-0-leaf
else node(T_1) is not leaf but node(T_2) is leaf
then create 4 branches for node(T_2): // case 5
node(T_2)-0-0-leaf, node(T_2)-0-0-leaf,
node(T_2)-0-0-leaf, and node(T_2)-0-0-leaf

// (else) case 6: both node(T_1) and node(T_2) are
// neither null nor a leaf, i.e., a branch
Contine the tree traversal using BFS/DFS down to tree T_1 as well as T_2 simutaneouly, and run the ARAR_Tree_Integration( ) on T_1's and T_2's corresponding nodes
End

Figure 17. Pseudo code for the integration of two ARAR trees

4 Diagnosis Visualization and System Performance Evaluation

Visual data analysis assists in perceiving patterns, trends, and exceptions in even the most complex data sources where visualization allows audience to identify concepts and relationships that they had not previously realized [11]. For the reason, the system prototype of our work has been developed and completed on the basis of our two tree-based diganosis mechanisms as well as visualized approach. Portions of the ACL configurations on the routers/firewalls in our campus network is put to use as the reference of the input of our system implementation. Figure 18 to Figure 21 showcase the system diagnosis results for the ACL rule configurations in our experimental network. Please notice that the implementation/visualization of our logical network topology is not described here since it is beyond the scope of this paper. In the upcoming future, we plan to integrate some well-known network management system, e.g., HP OpenView, to replace our current monitoring subsystem for a more complete system implementation.

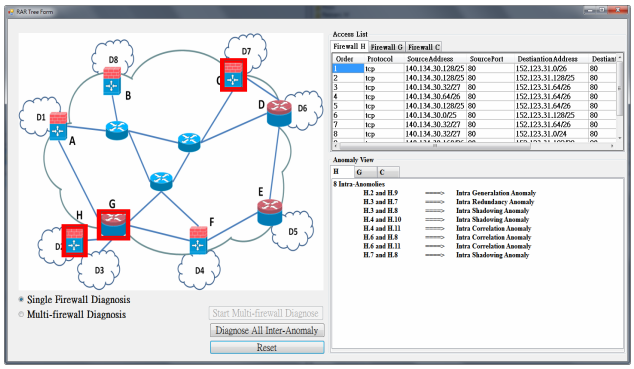

Figure 18. Intra-ACL anomalies on firewalls C, G, and H

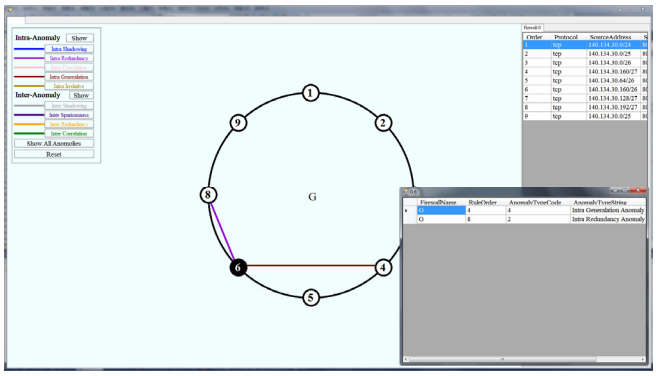

Figure 19. Insight of intra-ACL anomalies in firewall G

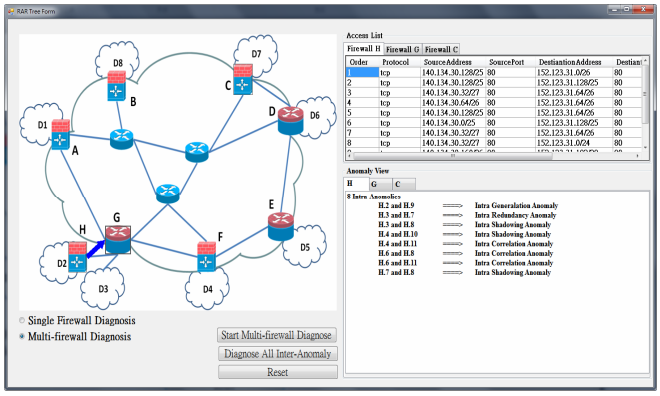

Figure 20. Inter-ACL anomalies between firewall H and firewall G

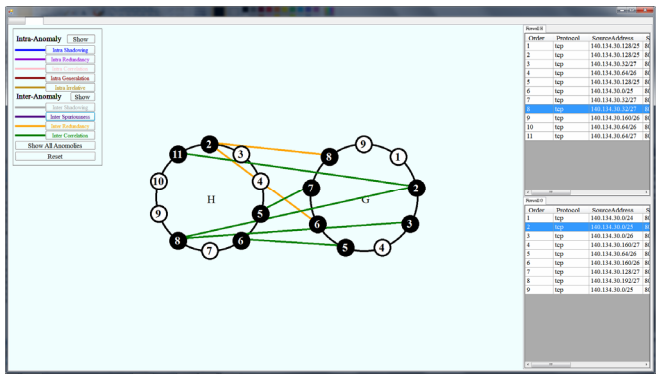

Figure 21. Insight of inter-ACL anomalies between firewall H and firewall G

To obtain the diagnosis results, users can first launch our Firewall Anomaly Diagnosis System (Figure 18) and the network topology is accompanied and shows up. Later, as in Figure 18, the user clicks the "Single Firewall Diagnosis" option on the window and our system would report that there are intra-ACL rule anomalies existing in firewalls C, G, and H. The network manager can click any of these three blinking icons on the network topology window to pop up the Intra-Anomaly View for a detailed look-up dedicated for the clicked firewall. In Figure 19, firewall G is clicked and the Intra-Anomaly View window is launched. The network manager can move the mouse cursor on top of the iconic circle of a rule to get the

anomaly relationship between the pointed rule and the others (Figure 19 shows the anomaly relationship for the rule numbered 6). Meanwhile, a text window pops up aside to show the textual configuration of these rules.

Likewise, users may select the "Multi-Firewall Diagnosis" option and click any two designed firewalls for the inspection of inter-ACL rule anomalies between them. Figure 20 shows if we first click firewall H and then firewall G on the network topology window for inter-ACL anomaly diagnosis, then there will be a directed link incident on these two iconic firewalls where the link orignates from the firewall H and is destined for Firewall G. Figure 21 shows the inter-ACL anomalies between these two firewalls which are displayed in our Inter-Anomaly View window. In Figure 21, the user can use the upper left panel to select which categories of inter-ACL anomalies he prefers to observe for a clearer view on the diagnosis results. Of course, the user can clink on any link indicating an inter-ACL anomaly between these two firewalls and the corresponding detailed configurations of the paired rules will show up at the right side of the window to allow the user realize contents of the rules causing the anomaly.

A comprehensive set of experiments had been conducted in our lab's networking environment to obtain the performance evaluation [9], where the ACL rules were constructed in a random fashion referring to [12], i.e., the IP range of each rule is generated on a random basis. The experimental results show our two diagnosis mechanisms are fully superior to that of [1], which needs a substantial amount of pair-wise rule comparisons to do anomaly diagnosis. Figure 22 and Figure 23 show the performance comparisons among our ARAR-tree-based system, the RAR-tree-based system, and the FSM-based diagnosis mechanism [1], where our RAR-tree-based system splits the traffic plane into fixed-sized smaller blocks (A is the size of blocks) and, in the case of Figure 22, rules have a larger size (and smaller size in Figure 23) of filtering area/space than A. It can be found that ARAR tree can always give us dramatic performance on rule anomaly diagnosis. As for diagnosis with FSM [1], the system execution time will grow exponentially with the growth of the number of rules such that we can not properly draw the growth line of its daignosis performance along with the other two systems in Figure 22 and Figure 23.

Among the diagnosis performance shown in Figure 22 and Figure 23, one another thing is worth noting: How do we chose the proper size of A (i.e. the size of split blocks) with our RAR tree-based mechanism while conducting diagnosis? In Figure 22, it can be found that diagnosis wih smaller split block sizes results in worse performance because the setting of larger split block sizes causes few fragments during RAR tree contstruction where rules have larger

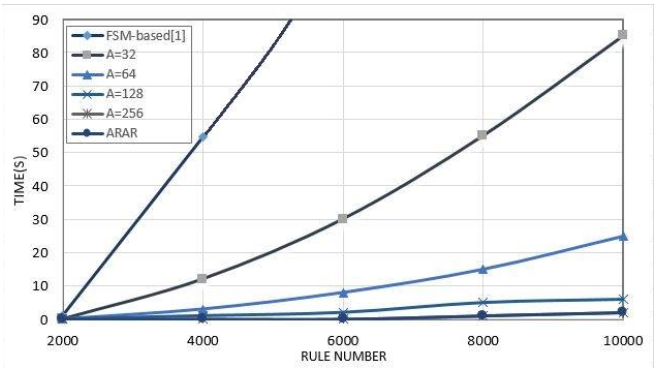

Figure 22. Diagnosis performance with larger rule filtering area

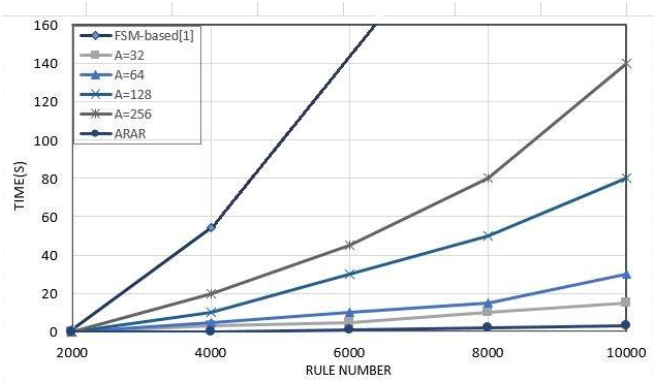

Figure 23. Diagnosis performance with smaller rule filtering area

filtering space. As a contrast, Figure 23 shows our RAR tree-based diagnosis with smaller split block sizes results in better performance where rules have a smaller filtering space. Considering the configuration uncertainty of rule filtering space, it is hard to choose a proper value of A prior to diagnosis to acquire better performance if our RAR tree-based mechanism is employed. It is the reason that our ARAR tree-based diagnosis version is developed.

As a result, in effectiveness, both of our tree-based systems are developed on the basis of National Institute of Standards & Technology (NIST) Special Publication 800-192 [13], which are derived from [1]. Comparing with the dagnosis method proposed by [1] in which a FSM-based pairwise rule comparison is employed, our two systems purge those uncessary comparisons between two rules which have no intersections with each other on filtering effects. Performance evaluations prove that our systems are far more effieient than [1] is.

Meanwhile, our two developed systems do the rule anomaly diagnosis at no expense of scalability. By re-using the individual local diagnosis results, our systems can effortlessly accomodate rule anomaly diagnosis among multiple firewalls, which can hardly be done at ease by any of the currently developed systems. As for usability, plenty of well-known tools with various visualized methodlogies have been

launched. Still, nowadays, none of them are deployed in reality [14], in which our work is also cited. To date, a prototype system developed with the collaboration with D-Link Co., Taiwan and based on our developed diagnosis mechanisms went live since early July this year, to facilitate the configuration and management of firewalls in our campus network.

5 Conclusion and Future Work

With implementations of the RAR tree as well as ARAR tree, both of our diagnosis mechanisms for firewall rule anomalies can meet the planned requirements: Effectiveness, efficiency, scalability, and usability. Shortening the time needed for the diagnosis of rule anomalies inside/among firewalls means reducing the possibilities of the loss of company estates, caused by network attacks. This is very important for those systems which run on-line and need speedy responses regularly with their users, e.g., on-line banking or online shopping. They tolerate no room for a second service break, leading to prompt and correct firewalls configuration in response to various threats coming from Internet.

Although we get a noticeable achievement on our system development, as the next steps, more interesting ingredients and plenty of technical challenges are in front of us, and expected to be delved into to complete our diagnosis system and meet the upcoming demands [15], e.g., migrating the current mechanism(s) to IPv6 networking environment, adding inspection functions for behavior mismatching among firewalls, developing next-version of systems for stateful firewall rules, and take port configuration/information into account. Meanwhile, a new data structure named E-ARAR (Enhanced ARAR) Tree and associated algorithms are devised [16] and tried to address the re-splitting issue during tree integration for inter-ACL rule anomaly diagnosis.

Acknowledgments

This work is supported by MOST, R.O.C., under contract MOST-104-2221-E-035-023.

Remark

This manusript is mainly revised and augmented from the versions of TANET 2010 as well as NCS 2013, Taiwan (both in Chinese).

References

[1] A. X. Liu, A. R. Khakpour, J. W. Hulst, Z. Ge, D. Pei, J. Wang, Firewall Fingerprinting and Denial of Firewalling Attacks, *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 7, pp. 1699-1712, July, 2017.

[2] E. Al-Shaer, J. Lobo, L. Kalger, *Policies for Distributed Systems and Networks*, IEEE Press, 2008.

[3] T. Wong, On the Usability of Firewall Configuration, *The 5th Symposium on Usable Privacy and Security*, Pittsburgh, PA, 2008, pp. 180-185.

[4] E. Al-Shaer, H. Hamed, R. Boutaba, M. Hasan, Conflict Classification and Analysis of Distributed Firewall Policies, *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 10, pp. 2069-2084, October, 2005.

[5] C.-S. Chao, A Visualized Internet Firewall Policy Validation System, *The 10th IEEE/IEICE Asia-Pacific Network Operations and Management Symposium*, Sapporo, Japan, 2007, pp. 364-374.

[6] Y. Yin, Y. Katayama, N. Takahashi, Detection of Conflicts Caused by a Combinations of Filters Based on Spatial Relationships, *Journal of Information Processing Society of Japan*, Vol. 49, No. 2, pp. 3121-3135, December, 2008.

[7] Y. Yin, R. S. Bhuvaneswaran, Y. Katayama, N. Takahashi, Implementation of Packet Filter Configurations Anomaly Detection System with SIERRA, *The 7th International Conference on Information, Communication and Signal Processing*, Bejing China, 2005, pp. 467–480.

[8] C.-S. Chao, M.-H. Yu, R.-Y. Pan, A RAR Tree-Based Diagnosis System for Rule Anomalies among Network Firewalls**,** *The 15th Taiwan Academic Network Symposium (TANET 2010)*, Taiwan, 2010, Session A2, No. 3.

[9] C.-S. Chao, C.-T. Chiu, An Adaptive RAR Tree-Based Diagnosis System for Rule Anomalies and Behavior Mismatching among Firewalls, *The 36th National Computer Symposium (NCS 2013)*, Taiwan, 2013, Session 3, No. 20.

[10] K. Sayood, *Introduction to Data Compression*, 3rd Ed., Elsevier, 2006.

[11]H. Shiravi, A. Shiravi, A. A. Ghorbani, A Survey of Visualization Systems for Network Security, *IEEE Transaction on Visualization and Computer Graphics*, Vol. 18, No. 8, pp. 1313-1329, August, 2012.

[12]T. Samak, A. El-Atawy, E. Al-Shaer, Towards Network Security Policy Generation for Configuration for Configuration Analysis and Testing, *The 2nd ACM Workshop on Assurable and Usable Security Configuration (SafeConfig' 09)*, New York, NY, 2009, pp. 13-18.

[13] NIST. SP. 800-192, Verification and Test Methods for Access Control Policies/Models, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-192.pdf, 2017.

[14] V. Artem, I. Leonardo, M. Leonardo, L. Stefan, Systematic Literature Review on Usability of Firewall Configuration, *ACM Computing Survey*, Vol. 50, Issue 6, No. 87, January 2018.

[15] E. Al-Shaer, *Automated Firewall Analytics: Design, Configuration and Optimization*, Springer, 2014.

[16] C.-S. Chao, H.-Y. Lin, An Efficient Anomaly Diagnosis Mechanism for Stateful Firewall Rules, *The 40th National Computer Symposium (NCS 2017)*, Hualien, Taiwan, 2017.

Biographies

Chi-Shih Chao currently is an associated professor at the Communications Engineering Dept. of Feng Chia University, Taiwan. His research interests include network security, network fault management, high-speed networks, and wireless LANs. Dr. Chao received the Annual Best Paper Awards from Taiwan *TANet* in 2015 and *IMP* in 2016, respectively. He also serves for plenty of relevant conferences, journals, and industrial committees. In addition, he is a member of IEEE and Phi-Tau-Phi.

Stephen J. H. Yang is the Vice President of Asia University, Taiwan. He is also associated with the National Central University as the Distinguished Professor of Department of Computer Science & Information Engineering. He received his Ph.D. degree in Electrical Engineering & Computer Science from the University of Illinois at Chicago in 1995. He has published over 60 SSCI/SCI journal papers, his research interests include big data, learning analytics, artificial intelligence, educational data mining, and MOOCs. He received the Outstanding Research Award from Ministry of Science & Technology (2010) and Distinguished Service Medal from Ministry of Education (2015).