Indian Journal of Science and Technology, Vol 9(47), DOI: 10.17485/ijst/2016/v9i47/101742, December 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Design of Tool to Detect Anomalies in Firewall Rules

Rakesh Yadav, Harjeet Kaur and Aman Saurabh

Department of Computer Science and Engineering, Lovely Professional University, India;
rakesh.yadav5705@hotmail.com,harjeet.kaur@lpu.co.in, amansaurabh10@gmail.com

Abstract

The role of a firewall is to accept or discard packets by examining them through a sequence of rules. Often these rules are conflicting and create anomalies. Managing firewall rules is somewhat complex. Effectiveness of any firewall depends upon the quality of policy configuration and its rule set. This paper describes an algorithm implemented in a tool which identifies the anomalies automatically in rule set by placing the new rule in its appropriate position. The presented approach is helpful in improving the efficiency of firewall and maintaining the appropriate order of firewall rule set to avoid anomalies.

Keywords: Anomalies, Firewall Rules, Policy, Policy Tree, Rule Relationships

1. Introduction

Firewall is nothing but the security guard which lies between the system and the internet. It is the most essential tool when a user is connected to the network and making communication on the network with the other remote devices or network devices. In order to implement security policy of network, firewall checks every incoming or outgoing packet and decides whether to accept it or discard, based on the set of rules defined by network administrator. The system security depends highly on these rules because if not configured appropriately, some undesired traffic may enter or may block the desired traffic. Each rule in a firewall is of the form[3].

<predicate> → <decision>

The <predicate> of a rule is a Boolean expression over some packet fields along with the physical network interface on which a packet arrives. The <decision> of a rule is either to accept, or to discard. If rules are defined manually, the probability of anomalies in rule set is more. Managing firewall policy is a challenging task and often leads to conflicting policy rules because defined rules are complex and interdependent. But the complexity of managing firewall policy limits the effectiveness of security provided by firewalls[2]. Situation becomes worse with increasing number of filtering rules. As number of rules are increased, generally in large scale enterprise network, the difficulty level of writing new rules, modifying the existing one also increases. A new rule added, may shadow some existing rule that is may hide the effect of some other rule. Thus creating anomalies in the rule set which significantly affect the security of the network.

The rest of the paper is structured as follows. Next section discusses the representation of firewall rules and types of anomalies in rule set. Main emphasis in section 3 is placed on exhibiting the design and implementation details of the incorporation of algorithm for anomaly discovery. Section 4 supports the whole discussion with experimental results to prove the effectiveness of the algorithm used and finally section 5 concludes the paper with future enhancements.

2. Firewall Rule Representation

Syntax of firewall rule can be represented as an ordered tuple containing certain fixed field as shown below

<Order><Protocol><Source_IP><Source_Port><Destination_IP><Destination_Port><Action>

Where *Order* is the position of the rule where it will be stored in the firewall rule set. Action may be either Accept or Reject for example Rule 1 placed at position 1 accepts packets sent from IP address 140.14.2.* from any port to any destination but at destination port 75 TCP while Rule 2 at position 6 does not allow communication between

*Author for correspondence

145.13.54.65 and 161.120.33.40 using UDP at destination port 75.

Rule 1 : <1, TCP, 140.14.2.*, Any, *.*.*.*, 75, Accept>

Rule 2 : <6, UDP, 145.13.54.65, Any, 161.120.33.40, 75, Deny>

2.1 Anomalies in Firewall Rules

Consider the following rule set:

Rule 1 : <1, UDP, 10.1.*.*, 53, 192.168.1.1, 80, ALLOW>

Rule 2 : <2, UDP, 192.168.25.*, 25, 168.142.1.2, 53, ALLOW>

Rule 3 : <3, UDP, 172.168.55.15, 80, 10.1.1.1, 80, ALLOW>

Rule 4 : <4, UDP, 10.1.1.*, 53, 192.168.1.1, 80, DENY>

Rule 5 : <5, UDP, 192.168.25.5, 25, 168.142.1.2, 53, DENY>

Rule 6 : <6, TCP, 192.168.*.*, 53, 172.142.1.2, 80, DENY>

Rule 7 : <7, TCP, 185.155.65.14, 53, 198.162.45.40, 80, DENY>

Rule 8 : <8, TCP, 192.168.*.*, 53, 172.142.1.2, 80, ALLOW>

Rule 9 : <9, TCP, 192.168.25.16, 53, 172.168.*.*, 53, ALLOW>

Rule 10 : <10, TCP, 10.1.1.*, 80, 192.168.1.*, 53, ALLOW >

Rule 11 : <11, TCP 192.168.25.*, 53, 172.168.4.5, 53, DENY>

Rule 12 : <12, TCP 10.1.1.*, 80, 192.168.1.*, 53, ALLOW>

Rule 3 and 7 are completely *disjoint* and will not lead to any kind of anomaly. *Shadowing* anomaly comes when one rule matches all the incoming packets and other rule does not get any chance to match the incoming packets. E.g. Rule 1 matches all the packets and does not pass the packets further to Rule 4 it means the rule 1 is shadowing to rule 4. And the rule 4 is never activated. In the presented rule set, shadowing anomaly also occurs between rule 6 and 8. Rule pair 1 and 4 is inclusively matching while the pair 6 and 8 is exact match. Rule 9 and Rule 11 are correlated as both have different actions for same packet. Rule 11 is denying all packets from a particular source to destination at all ports while contradicting to this is Rule 9 which wants this communication to happen at destination port 53. This is *correlation* anomaly where incoming packet is matched by two different rules and both the rules have different actions. A rule is said to be *redundant* if a similar rule exists in the rule set that matches the same packet and perform the same action. If Rule 10 is removed, rule set will not be affected as Rule 12 is performing the same action on same packets. Existence of redundant rules takes additional space and degrades the performance of firewall. Apart from these, some other anomalies have also been defined. *Generalization* anomaly is for rules in order, making different action. But if the order of the rule is reversed, action will change and superset rule will shadow the other rule as in Rule 2 and Rule 5. The superset rule is known as the General rule[7]. If at a desired interval of time the rule does not match any packet then it is known as *Irrelevance* anomaly. Anomalies generally occurs while rule updating[8] where positions of the rules get distributed.

But manually it is very difficult to detect the anomalies and also to resolve it be-cause a firewall is having thousands of rules. And still the irrelevance anomaly is not detected because of time constraints among the rules[4]. Security policy of any firewall is totally dependent on its defined rule set. Protection level is increased by defining strong rules, hundreds or can say thousands of rules are defined to make an effective firewall system. Appropriate positioning of rules in such a large rule set is the biggest challenge for network administrator and directly affects the security of the network. Generally the rules are custom designed and hand written. If not defined and maintained carefully will allow unwanted traffic to enter into the network or deny passage to required packets. Manual definition and maintenance of rules is complex, error prone, costly and inefficient. Things become complex with the size of rule set. As more rules are added, chances of getting anomalies also increases and rule management becomes difficult. It may leads to create erroneous rules with-in the set of rules and don't allow network to not perform according to the service. If these errors occur then the anomalies are created which are to be detected & removed from the firewall rule set.

2.2 Rule Relation

Anomaly detection algorithm requires rules to be compared. For rule comparison set relations are used as each rule is an ordered tuple. As far as this work is concerned, relationship between two rules can be – Disjoint, Exactly Matching, Inclusively Matching or Correlated[1]. Two rules

are fully *disjoint* if they are having at least one parameter in rule representation. Rule 3 and Rule 7 are completely disjoint as both are meant for entirely different destination. A rule r1 is said to be *inclusively matchi*ng with r2, if r1 is subset or r2 or there is at least one parameter for which value in r1 is subset of value in r2 and rest of the parameters are same. Rules have *exact match* if an already existing rule is added to rule set as new rule.

3. Methods/Statistical Analysis

Algorithm has been divided into two modules – Anomaly detection and its resolution. The new rule set is anomaly free.

3.1 Anomaly Detection

All anomalies are somehow related with corresponding rule relations. Table 1 defines anomalies in terms of rule relations.

Table 1. Anomalies in terms of rule relation

Anomalies	Firewall Rule Relation
Shadowing Anomaly	Inclusive or Exact Match
Correlation Anomaly	Correlated Match
Redundancy Anomaly	Redundant Match
Generalization Anomaly	Partially Disjoint

Figure 1 exhibits the rule relationship and corresponding anomaly in the form of a state diagram. Shadowing anomaly between two rules x and y will be detected

```
if (Proto_y == Proto_x)
  if (Src_y == Src_x)
     if (isSubset (Dsty,Dst_x) )
        { ..
           Shadowing Anomaly
          ..
         }
       else
        { ..
           Redundant Anomaly
          ..
}
```

3.2 Anomaly Resolution

After detecting anomalies between two rule sets, one rule has to be discarded. The decision is to be made in such a way that it further does not lead to any other anomaly[5]. Table 2 presents the solution to be followed if two rules are having anomalies as classified in previous section.

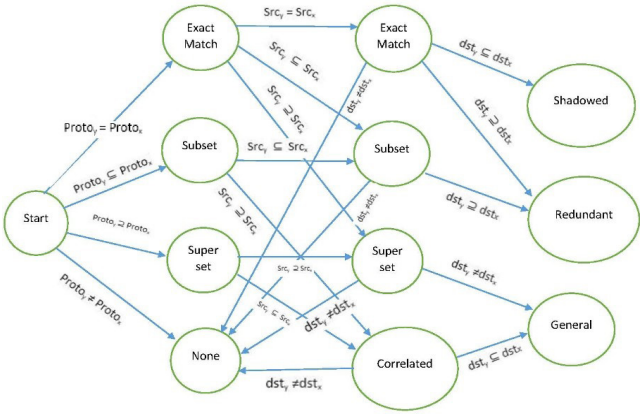

Figure 1. State Diagram for detecting anomalies in two rule sets

Table 2. Anomaly Resolution

Anomalies	Solution
Shadowing Anomaly (Exact Match)	Discard rule with ALLOW action
Shadowing Anomaly (Inclusive Match)	Bring subset rule before superset rule
Correlation Anomaly	Break the rules into disjoint parts and reinsert them in rule set
Redundancy Anomaly	Remove the redundant rule

3.3 Rule Insertion

From an existing rule set, a completely new rule set is made by comparing each rule with all other existing rules[6]. If any inconsistency is detected, it is resolved and new or modified rules are added into new rule set. Using the states of diagram in Figure 1 for detection and solutions in Table 2, the rule set presented in section 2.1 can be represented in the form of policy tree (Figure 2(a) and (b)) which places the rules in their appropriate anomaly free position.

4. Findings

The concept of anomaly detection and resolution has been incorporated in a user friendly tool. Tool has been designed to serve both categories of users – Regular and Admin. It contains options for file encryption/decryption, rule engine design, rule generation and updating of con-

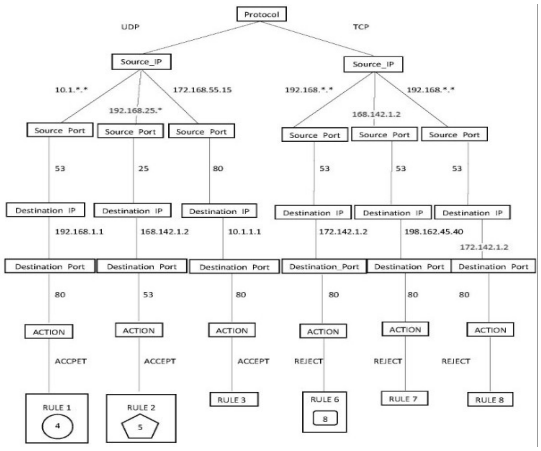

Figure 2 (a). Policy tree

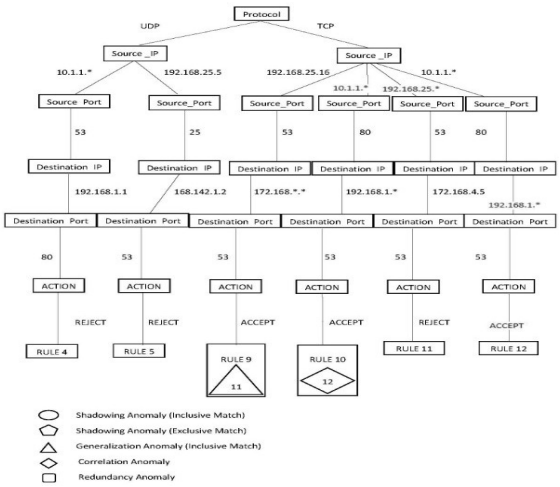

Figure 2 (b). Policy tree

flicting rules. It can be used as rule advisor for identifying the conflicted rules, shadowing, redundant and correlation anomalies. As rule editor it provides the facility for rule insertion, modification and deletion. When the new rule is inserted it automatically finds the correct position in the rule set. A snapshot of the designed tool is shown in Figure 3(a) and (b).

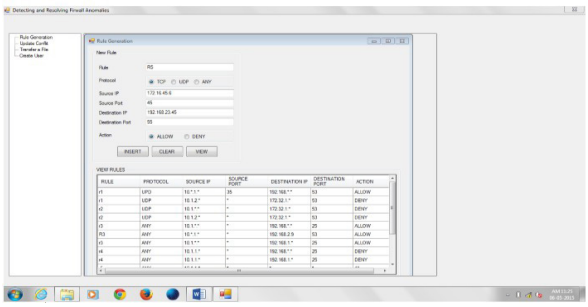

Figure 3 (a). Rule Generation interface

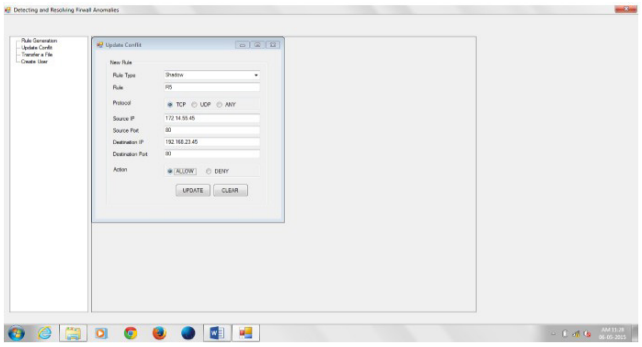

Figure 3 (b). Update Conflicting Rules

5. Conclusion and Future Scope

Tool has been designed to automate the anomaly detection process in firewall rule set which if done manually is time consuming, require expertise and sometimes may lead to some other type of anomaly. It also incorporates the anomaly resolution and gives interface to update conflicting rules. The work can be improved by considering the efficiency of working of tool with large rule set.

6. References

1. Ehab S. Al-Shaer, Hazem H. Hamed. DePaul University, Modeling and Management of Firewall Policies. 2004 IEEE, *eTransactions on Network and Service Management*, Second Quarter, 2004
2. Mohsen Rezvani, Ramtin Aryan. Department of Information Technology and Computer Engineering, Shahrood University of Technology, Shahrood, Iran. Specification, Analysis and Resolution of Anomalies in Firewall Security Policies. *World Applies Science Journal.* 7 (Special Issue of Computer & IT): 188-198, 2009 ISSN 1818.4952 Copyright IDOSI Publications, 2009.
3. Mohamed G Gouda, Alex X Liu. Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712-0233, United States, *Structured Firewall Design*.
4. Rupali Chaure, Shishir K Shandilya. Firewall anomalies detection and removal techniques – a survey. *International Journal on Emerging Technologies*. 2010; 1(1):71-74.
5. Hongxin Hu, Grail-Joon Ahn, Ketan Kulkarni. Arizona State University Tempe, AZ 85287, USA, FAME: A Firewall Anomaly Management Environment. *Safe Config'10, October 4*, 2010, Chicago, Illinois, USA.
6. Hongxin Hu, Grail-Joon Ahn, Kulkarni Ketan. Members IEEE, Detecting and Resolving Firewall Policy Anomalies. *IEEE Transactions on Dependable and Secure Computing.* 2012 May/Jun; 9(3).

7. Ganesh Sriram G, Swarna Sri BI, Rama Devi N, Gouthami K, Arjun Kumar S. Dept. of CSE, Srinivasa Institute of Engineering & Tech, NH-216, Cheyyeru, AP, India "Anomalies Detection and Recovery in Firewall Policies" *IJCST*. 2013 Jan-Mar; 4(1).
8. Jayabalan Lekha, Ganapathi Padmavati. Department of Computer Science, Avinashiligam Institute for Home Science and Higher Education for Women, Coimbatore – 6411008, Tamil Nadu, India, *INDJST*. 2016 Apr; 9(13). DOI: 10.17485/ijst/2016/v9i13/87983. Anomaly based Malicious Traffic Identification using Kernel Extreme Machine Learning (KELM) Classifier and Kernel Principal Component Analysis (KPCA).