International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 14 (2015) pp 34752-34755

Coherent Translucent Testing Of Firewall Policies

R.RENUGA DEVI

Mail id: renuinfotech23@gmail.com, Mobile: 9159537623
Teaching Fellow, Anna University-BIT Campus-Trichy-620 024

ABSTRACT:
Firewall is used to control the incoming and outgoing network traffic by analyzing the data packets and determining whether it should be allowed through or not based on predefined rule set. The quality of protection provided by a firewall is done by ensuring the correctness of its Policies, which makes the quality an important factor in firewall Policy and yet difficult. To ensure the correctness of firewall policy a logical transparent testing approach is proposed. Logical transparent testing uses structural coverage criteria. Structural coverage criteria of rules, predicates, and clauses are proposed to test the firewall policy. Also it is planned to use four automated packet generation techniques, such as the random packet generation, local constraint solving, global constraint solving, and the boundary value analysis for achieving high structural coverage Test reduction technique also proposes to reduce the fault detection capability. Finally to detect faulty generated packet set, an experiment on a set of real policies and a set of faulty policies will be done. The detected faults are planned to maintain in a log for identifying the fault which has occurred frequently with the help of association mining rule.

Key Notes: Transparent testing, Firewall policies, Firewall policy testing.

INTRODUCTION

A system or group of systems that enforces an access control policy between two networks is called a firewall. It implements a network access policy by forcing connections to pass through it, where they can be examined and evaluated. A firewall is a fundamental asset of network security. A cornerstone of a firewall's functionality is its capability for implementing and enforcing a network access policy[2]. This policy consists of a set of conditions represented by a list of rules. In a distributed system, messages are encapsulated into packets, which often pass through multiple access points in a network and firewalls are responsible for filtering, monitoring, and securing such packets. Firewalls are often placed at the perimeter of a network. Such a firewall can be said to have an external and internal interface, with the external interface being the one on the outside of the network. These two interfaces are sometimes referred to as unprotected and protected, respectively. Firewalls are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet[5]. All data entering or leaving the Intranet pass through the firewall, which examines each packet and blocks those that do not meet the specified security criteria. There are different types of firewalls depending on where the communication is taking place, where the communication is intercepted and the state that is being traced.

1. Packet filtering firewall
2. Application-layer firewall
3. Stateful packet inspection firewall

FIREWALL CONFIGURATION

A firewall policy is composed of a sequence of rules that specify under what conditions a packet is accepted and discarded while passing between a private network and the outside Internet. In other words, the policy describes a sequence of rules to decide whether packets are accepted (or) discarded. A rule is composed of a set of fields generally including source/destination IP addresses, source/destination port numbers, and protocol type and a decision. Each field represents the range of possible values to match the corresponding value of a packet, which is either a single value or a finite interval of non-negative integers[4]. A packet matches a rule if and only if each value of the packet satisfies the corresponding values in the rule. Upon finding a matching rule, the corresponding decision of that rule is derived. When evaluating a packet, the firewall policy follows the first-match semantic: the first matching rule is given the highest priority among all the matching rules.

CONFIGURATION MODEL

As a basic requirement of firewall policy management is modeled solutions, the relations and the representation of firewall rules in the policy. Rule relation modeling is necessary for analyzing firewall policy and designing management techniques such as conflict detection and rules editing. The rules or policy representation modeling is important for implementing these management techniques and visualizing the firewall policy structure. In describe formally model of firewall rule relations and policies. The **<predicate>** of a rule is a boolean expression over some packet fields such as source IP address, destination IP address, source port number, destination port number, and protocol type. The **<decision>** of a rule can be accept, discard, or a combination of these decisions with other options such as a logging option. as the evaluation result when the <predicate> is evaluated to be true. In a policy model, represents a value in a field fi (e.g., IP address) and represents its corresponding range of S_i (e.g., fi ∈ [2, 5]) to simplify the representation format. The first-match semantic of a firewall policy shows the same behavior with the execution of a series of IF-THEN-ELSE statements in program code.

SYSTEM DESIGN

When evaluating a packet, the firewall policy follows the first-match semantic: the first matching rule is given the highest priority among all the matching rules. In firewall testing focus on testing to cover only specific entities based on a set of defined coverage criteria. A test suite is a set of packet-decision pairs to check whether a packet is evaluated to its corresponding expected decision. The structural coverage measurements are monitoring whether rules, predicates, or clauses are covered when evaluating packets against the policy under test. Policy testers may generate and select a test suite to achieve a certain (high) level of coverage. However, the main objective, through testing, is to detect faults in the firewall policy while reaching a certain level of coverage. Coverage analysis helps investigate a larger portion of entities for fault detection using a test suite that achieves higher structural coverage. Each module is described in the rest of the section:

1. Firewall Policies
2. Structural Coverage Criteria
3. Test Packet Generation
4. Test Reduction
5. Fault Detection and Log maintenance.

Firewall Policies

The Firewall policy describes a sequence of rules to decide whether packets are accepted or discarded. A rule is composed of a set of fields (generally including source/destination IP addresses, source/destination port numbers, and protocol type) and a decision. Each field represents the range of possible values (to match the corresponding value of a packet), which are either a single value or a finite interval of non-negative integers. A packet matches a rule if and only if each value of the packet satisfies the corresponding values in the rule. Upon finding a matching rule, the corresponding decision of that rule is derived. A firewall policy is based on common generic features. A firewall policy is composed of a sequence of rules, each of which has the form called the generic representation.

Structural Coverage Criteria

A test suite is a set of packet-decision pairs to check whether a packet is evaluated to its corresponding expected decision. The structural coverage criteria contains three different criteria for evaluate the packets. Those criteria's are follows.

Rule Coverage Criteria (RCC)

RCC for a test suite requires that for each rule r in a policy, the evaluation of the test packets in the test suite needs to match r at least once, respectively. For the rule coverage criterion, rule coverage is the percentage of the number of covered rules in a policy.

Predicate Coverage Criteria (PCC)

PCC for a test suite requires that for each predicate p of the rules in a policy, the evaluation of the test packets in the test suite needs to make p to be evaluated to true and false at least once, respectively. For the predicate coverage criterion, predicate coverage is the percentage of the number of covered predicates (i.e., predicates being evaluated to true or false) in P over 2 × |P|.

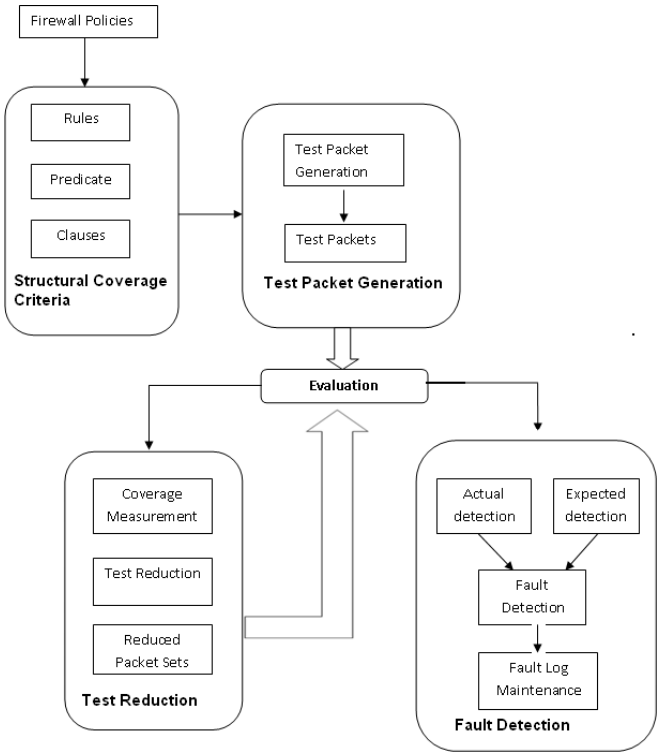

Figure System Architecture for Transparent Testing

Clauses Coverage Criteria (CCC)

CCC for a test suite requires that for each clause c of the predicates in a policy, the evaluation of the test packets in the test suite needs to make c to be evaluated to true and false at least once, respectively. For the clause coverage criterion, clause coverage is the percentage of the number of covered true or false values of clauses in C over 2 × |C|.

Test Packet generation

The main aim of this module is to generate the packet for achieve high structural coverage and send the packets into the firewall policy. Based on the result the packets are evaluated. In this section describes four packet generation techniques.

Random Packet Generation

This technique does not require the policy itself in test generation and can quickly generate a large number of test packets; the technique often lacks effectiveness to achieve high structural coverage with the generated packets. Due to randomness, the number of the entities (i.e., predicates or clauses) being covered is often small in comparison to the total number of the entities in the policy under test.

Global Constraint Solving

Global constraint solving is to take into account the influence overlapping predicates across the rules. Covering entities in a rule requires that the predicates of all the receding rules should be evaluated to false. This technique uses a Rule reach ability definition. Rules reach ability of a packet k to reach a rule ri in a policy requires that k evaluates ri's preceding rules' predicates to false and reaches the rule.

Boundary Value Analysis

The generated packets include boundary values, which are on the range boundaries (i.e., the smallest value and the largest value) of each field. Intuitively, when a fault is injected to a firewall policy, it is likely that the policy includes a faulty policy behavior on range boundaries of a field instead of other values. The technique selects boundary values instead of random values from values satisfying rule constraints. Boundary values are the values around the smallest and largest values of a clause in a rule.

Test Reduction

The pervious module generates more number of packets. But the test suite requires minimum number of packet decision. So the test reduction is needed. Based on the inspection of packets without incurring substantial loss in fault detection capability the size has reduced. Since structural coverage is an important factor for reflecting fault detection capability, while keeping its coverage level the size can reduce in the test suite. This technique uses Greedy algorithm for remove a packets from the packet set if and only if evaluating the packet does not increase any of the coverage metrics that are achieved by previously evaluated packets in the packet set.

Fault Detection and Log maintenance

Fault detection is a focus of any testing process. The main aim is to investigate the relationship between firewall policy structural coverage achieved by a packet set and the packet set's fault-detection capability. The mutation testing is adopted to measure the fault-detection capability of the packet set. In policy mutation testing, we inject a fault into the original policy and thereby create a mutant. Injected faults can be of various types including simple mistakes and complex configuration errors involving multiple rules. The fault log maintenance system is to maintain the detecting the faults in this module. This module provides the frequent occurrences of fault details with help of association mining rule.

CONCLUSION AND FUTURE WORK

The logical transparent testing of firewall policies has been obtained after a detailed study of more than ten IEEE transactions that involves conference within it. The survey being done has mainly lead to the identification of the logical transparent testing of firewall policies involves rules, Predicates, Clauses that achieve global constraint solving and boundary value analysis. A packet with high structural coverage has higher fault detection capability that detects more injected faults. The inbuilt fault log maintenance system provides an efficient backend that picks up frequent fault using association rule mining. The system has lead to the maintenance of similar fault detection capability with original set under a reduced packet set.

REFERENCES

[1] A. Wool, "A quantitative study of firewall configuration errors," Computer, vol. 37, no. 6, pp. 62-67, 2004.

[2] E. Al-Shaer and H. Hamed, "Discovery of policy anomalies in distributed firewalls," in Proc. 2004 IEEE Conf. on Communications, pp. 2605-2616.

[3] L. Yuan, J. Mai, Z. Su, H. Chen, C.-N. Chuah, and P. Mohapatra,"FIREMAN: a toolkit for FIREwall Modeling and ANalysis," in Proc.2006 IEEE Symposium on Security and Privacy, pp. 199-213.

[4] M. R. Lyu and L. K. Y. Lau, "Firewall security: policies, testing and performance evaluation," in Proc. 2000 International Conference on Computer Systems and Applications, pp. 116-121.

[5] P. Ammann, J. Offutt, and H. Huang, "Coverage criteria for logical expressions," in Proc.2003 International Symposium on Software Reliability Engineering, pp.99-107.

[6] A. X. Liu and M. G. Gouda, "Complete redundancy detection in firewalls," in Proc 2005 Annual IFIP Conference on Data and Applications Security, pp. 196-209.

[7] E. Martin, T. Xie, and T. Yu, "Defining and measuring policy coverage in testing access control policies," in Proc. 2006 International Conference on Information and Communications Security, pp. 139-158.

[8] P. E. Black, V. Okun, and Y. Yesha, "Mutation operators for specifications," in Proc. 2000 IEEE International Conference on Automated Software Engineering, pp. 81-88.

[9] J. Jurjens and G. Wimmel, "Specification-based testing of firewalls," in Proc. 2001 International Andrei Ershov Memorial Conference on Perspectives of System Informatics, pp. 308-316.

[10] E. Martin and T. Xie, "A fault model and mutation testing of access control Policies," in Proc. 2007 International Conference on World Wide Web, pp. 667-676.

[11] A. El-Atawy, T. Samak, Z. Wali, E. Al-Shaer, F. Lin, C. Pham, and S. Li, "An automated framework for validating firewall policy enforcement," in Proc. 2007 IEEE International Workshop on Policies for Distributed Systems and Networks, pp. 151-160.

[12] A. X. Liu, M. G. Gouda, H. H. Ma, and A. H. Ngu, "Non-intrusive testing of firewalls," in Proc. 2004 International Computer Engineering Conference, pp. 196-201.

[13] D. Hoffman and K. Yoo, "Blowtorch: a framework for firewall test automation," in Proc. 2005

IEEE/ACM international Conference on Automated Software Engineering, pp. 96-103.

[14] H. Zhu, P. A. V. Hall, and J. H. R. May, "Software unit test coverage and adequacy," ACM Computer. Survey. vol. 29, no. 4, pp. 366-427, 1997.