Change-Impact Analysis of Firewall Policies

Alex X. Liu

Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824-1266, U.S.A
alexliu@cse.msu.edu

Abstract. Firewalls are the mainstay of enterprise security and the most widely adopted technology for protecting private networks. The quality of protection provided by a firewall directly depends on the quality of its policy (i.e., configuration). Due to the lack of tools for analyzing firewall policies, most firewalls on the Internet have been plagued with policy errors. A firewall policy error either creates security holes that will allow malicious traffic to sneak into a private network or blocks legitimate traffic and disrupts normal business processes, which in turn could lead to irreparable, if not tragic, consequences.

A major source of policy errors stem from policy changes. Firewall policies often need to be changed as networks evolve and new threats emerge. In this paper, we first present the theory and algorithms for firewall policy change-impact analysis. Our algorithms take as input a firewall policy and a proposed change, then output the accurate impact of the change. Thus, a firewall administrator can verify a proposed change before committing it.

1   Introduction

Serving as the first line of defense against malicious attacks and unauthorized traffic, firewalls are cornerstones of network security and have been widely deployed in businesses and institutions. A firewall is placed at the point of entry between a private network and the outside Internet such that all incoming and outgoing packets have to pass through it. The function of a firewall is to examine every incoming or outgoing packet and decide whether to accept or discard it. This function is specified by a sequence (i.e., an ordered list) of rules, which is called the "policy", *i.e.*, the configuration, of the firewall. Each rule in a firewall policy is of the form $\langle predicate \rangle \rightarrow \langle decision \rangle$. The $\langle predicate \rangle$ 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 $\langle decision \rangle$ of a rule can be *accept*, *discard*, or a combination of these decisions with other options such as a logging option. The rules in a firewall policy often conflict. To resolve such conflicts, the decision for each packet is the decision of the first (i.e., highest priority) rule that the packet matches. Table 1 shows an example firewall.

Table 1. An example firewall

Rule	Source IP	Destination IP	Source Port	Destination Port	Protocol	Action
r_1	*	192.168.0.1	*	25	TCP	accept
r_2	1.2.3.4	*	*	*	*	discard
r_3	*	*	*	*	*	accept

Although a firewall policy is a mere sequence of rules, correctly maintaining one is by no means easy. First, the rules in a firewall policy are logically entangled because of conflicts among rules and the resulting order sensitivity. Second, a firewall policy may consist of a large number of rules. A firewall on the Internet may consist of hundreds or even a few thousand rules in extreme cases. Last but not least, an enterprise firewall policy often consists of legacy rules that are written by different administrators, at different times, and for different reasons, which makes maintaining firewall policies even more difficult. Analyzing a large and complex sequence of logically related rules is certainly beyond human capability. Effective methods and tools for analyzing firewall policies, therefore, are crucial to the success of firewalls. However, firewall administrators are woefully under-assisted due to the lack of firewall policy analysis tools. Quantitative studies have shown that most firewalls on the Internet are plagued with policy errors [1]. A firewall policy error either creates security holes that will allow malicious traffic to sneak into a private network or blocks legitimate traffic and disrupts normal business processes, which in turn could lead to irreparable, if not tragic, consequences.

1.1   Motivations

Firewall policies are always subject to change due to a variety of reasons. Making policy changes is a major task of firewall administrators. For example, new network threats such as worms and viruses may emerge. To protect a private network from new attacks, firewall policies need to be changed accordingly. Modern organizations also continually transform their network infrastructure to maintain their competitive edge by adding new servers, installing new software and services, expanding connectivity, etc. In accordance with network changes, firewall policies need to be changed as well to provide necessary protection.

Unfortunately, making changes is a major source of firewall policy errors. Making correct firewall policy changes is remarkably difficult due to the interleaving nature of firewall rules. For example, when a firewall administrator inserts a new rule into a firewall policy, the meaning of the rules listed under this rule could be incorrectly changed without being noticed. Furthermore, firewall policy changes are made by human administrators, and it is common that human administrators make mistakes. It has been shown that administrator errors are the largest cause of failure for Internet services, and policy errors are the largest category of administrator errors [2].

Firewall policy errors can be dangerous and costly. On one hand, if a firewall policy error permits illegitimate communication, outside attackers may use these security holes to launch attacks. On the other hand, if a firewall policy error disallows legitimate communication, it may cause significant loss due to interrupted business. For example, if a firewall policy error prevents the communication between a web server and its supporting database server, all transactions that need such communication are disrupted.

1.2   The Problem

The fundamental problem in changing firewall policies is, how does a firewall administrator know that the change made to the firewall policies is correct? For example, suppose a firewall administrator wants to make a change to the firewall policy to allow a database server to talk to a web server. How does the administrator know that the change indeed enables this communication? Also, how does the administrator know that the change does not allow some other illegitimate traffic to flow as a side effect, given the subtle behavior of firewall rules? Such questions are exceptionally difficult to answer given the high complexity of firewall rules.

In this context, if there is a tool that takes a firewall configuration and a proposed change as input, then outputs the precise impact of the change, the errors caused by making policy changes would be greatly reduced. The impact of a change shows all the traffic that was formerly discarded, but is now accepted, and all the traffic that was formerly accepted, but is now discarded. The output impact must be human readable. With this tool on hand, a firewall administrator can examine the change-impact for unintended consequences.

1.3   Key Contributions

In this paper, we make the following three key contributions.

1. We develop a theory for firewall policy change-impact analysis. We identify four types of firewall policy changes: rule deletion, rule insertion, rule modification, and rule swap. For each type of change, we have a theorem that states the decisions of what packets will be changed due to the policy change. These theorems serve as the foundation for developing algorithms for computing firewall policy change-impact.
2. We present algorithms for firewall policy change-impact analysis. The input of our algorithms includes a firewall policy and a proposed change, and the output is the accurate impact of the change. Using our algorithms, an administrator can verify a proposed change before committing it.
3. We present a way to correlate the impact of a firewall policy change and the high level security requirements that the firewall needs to satisfy. We also present methods for making corrections if the impact of a change is not desirable.

Because the focus of this paper is on security policies, we simply use the term "firewall" to mean "firewall policy", "firewall rule set", or "firewall configuration" unless otherwise specified.

1.4   Road Map

The rest of this paper proceeds as follows. In Section 2, we show an example application of our firewall change-impact analysis tool. In Section 3, we present the theory foundation for firewall change-impact analysis. Based on these theorems, we develop algorithms for computing the impact of firewall changes in Section 4. In Section 5, we discuss some further issues for firewall change-impact analysis. In Section 6, we examine previous work and compare it with our approach. In Section 7, we give concluding remarks.

2   Example

In this section, we show an example application of our firewall policy change-impact analysis tool. Consider the example firewall in Table 1. We suppose that the private network behind this firewall has a mail server and a web server, whose IP addresses are 192.168.0.1 and 192.168.0.2 respectively. We further suppose that this firewall is required by its high level security policies to discard all packets from a malicious host whose IP address is 1.2.3.4.

Here we briefly explain the meaning of the three rules in Table 1. Rule r_1 means that all email packets to the email server are accepted. Note that for a packet, if its destination port number is 25 and its protocol type is TCP, then the packet is an email (SMTP) packet. Rule r_2 means that all packets from 1.2.3.4 are discarded. Rule r_3 means that all packets are accepted. Note that whenever a packet arrives at a firewall, the decision of the first rule that the packet matches is executed.

2.1   Rule Deletion

Suppose that the administrator of this firewall wants to delete rule r_1. Our change-impact analysis tool will output the following impact as shown in Table 2. The meaning of the impact is as follows: for the email packets from the malicious host 1.2.3.4 to the email server 192.168.0.1, before deleting rule r_1, the decision for such packets is *accept*; after deleting rule r_1, the decision for such packets is *discard*.

2.2   Rule Insertion

Suppose that the administrator of this firewall wants to insert the following rule above rule r_1:

Src IP	Dest. IP	Src Port	Dest. Port	Protocol	Action
*	192.168.0.2	*	80	TCP	accept

The meaning of this new rule is to accept all the HTTP packets to the web server 192.168.0.2. After the administrator gives this intended change and the original firewall in Table 1 to our change-impact analysis tool, the tool will output the following impact:

Table 2. Impact after deleting r_1 from the firewall in Table 1

Source IP	1.2.3.4
Destination IP:	192.168.0.1
Source Port:	*
Destination Port:	25
Protocol Type:	TCP
Decision before change:	accept
Decision after change:	discard

Table 3. Impact after inserting a rule above r_1 in the firewall in Table 1

Source IP	1.2.3.4
Destination IP:	192.168.0.2
Source Port:	*
Destination Port:	80
Protocol Type:	TCP
Decision before change:	discard
Decision after change:	accept

2.3   Rule Modification

Suppose that the administrator of this firewall wants to modify rule r_1 to be the following rule:

Src IP	Dest. IP	Src Port	Dest. Port	Protocol	Action
*	192.168.0.1	*	*	TCP	accept

The meaning of this modified rule is to accept all the TCP packets to the mail server 192.168.0.1. For this intended change, our change-impact analysis tool outputs the following impact:

2.4   Rule Swap

Suppose that the administrator of this firewall wants to swap rule r_1 and r_2. Similarly, for this intended change, our change-impact analysis tool outputs the same impact as shown in Table 2.

3   Change-Impact Analysis: Theorems

In this paper, we consider the following four types of changes that one can make to a firewall $\langle r_1, \cdots, r_n \rangle$. Recall that the predicate of the last rule in a firewall is always a tautology, which is for the purpose of ensuring the comprehensiveness property of the firewall. Therefore, we assume that one does not change the last rule. (Actually, given any firewall where the predicate of the last rule is not a

Table 4. Impact after modifying r_1 in the firewall in Table 1

Source IP	1.2.3.4
Destination IP:	192.168.0.1
Source Port:	*
Destination Port:	[1,24]
Protocol Type:	TCP
Decision before change:	discard
Decision after change:	accept
Source IP	1.2.3.4
Destination IP:	192.168.0.1
Source Port:	*
Destination Port:	[25, 65536]
Protocol Type:	TCP
Decision before change:	discard
Decision after change:	accept

tautology, we can modify the predicate of the last rule to be a tautology without changing the semantics of the firewall. Due to space limitation, we omit the proof.)

1. Deletion: delete rule r_i, where $1 \leq i \leq n - 1$.
2. Insertion: insert a new rule r between r_i and r_{i+1}, where $1 \leq i \leq n - 1$.
3. Modification: modify rule r_i to be r_i', where $1 \leq i \leq n - 1$.
4. Swap: swap the two rules r_i and r_j, where $1 \leq i < j \leq n - 1$.

Each rule in a firewall is associated with two sets of packets, a matching set and a resolving set [3]. More precisely, consider a firewall f that consists of n rules $\langle r_1, r_2, \cdots, r_n \rangle$. The matching set of a rule r_i, denoted $M(r_i)$, is the set of all packets that match r_i. The resolving set of a rule r_i, denoted $R(r_i, f)$, in firewall f is the set of all packets that match r_i, but do not match any r_j $(j < i)$ that is listed before r_i in f. The essence of the resolving set $R(r_i, f)$ of a rule r_i in firewall f is: for any packet p in $R(r_i, f)$, the decision of firewall f for packet p is the decision of rule r_i. Note that the matching set of a rule depends only on the rule itself, while the resolving set of a rule depends on both the rule itself and all the rules listed before it in a firewall.

3.1   Theory Foundation

The following four theorems lay the foundation for computing firewall change-impact. In this paper, we use $r.D$ to denote the decision of rule r, and $f(p)$ to denote the decision of the first (i.e., highest priority) rule that p matches in firewall f.

Theorem 1 (Rule Deletion Theorem). *Let f be a given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we delete rule r_i where $1 \leq i \leq n - 1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_{i-1}, r_{i+1}, \cdots, r_n \rangle$. We use g to denote the firewall $\langle r_{i+1}, \cdots, r_n \rangle$, which consists of the $n - i$ rules $r_{i+1}, \cdots, r_n$ after rule r_i in firewall f. For any packet p in Σ, consider the following two cases:*

1. if $p \in R(r_i, f)$, then $f(p) = r_i.D$ and $f'(p) = g(p)$, which means that f and f' may have different decisions for p;
2. if $p \in \Sigma - R(r_i, f)$, then $f(p) = f'(p)$, which means that f and f' have the same decision for p. □

Theorem 2 (Rule Insertion Theorem). *Let f be the given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we insert a new rule r between r_i and r_{i+1} where $1 \le i \le n-1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_i, r, r_{i+1}, \cdots, r_n \rangle$. We use g to denote the firewall $\langle r_{i+1}, \cdots, r_n \rangle$, which consists of the $n - i$ rules $r_{i+1}, \cdots, r_n$ after rule r_i in firewall f. For any packet p in Σ, consider the following two cases:*

1. if $p \in R(r, f')$, then $f(p) = g(p)$ and $f'(p) = r.D$, which means that f and f' may have different decisions for p;
2. if $p \in \Sigma - R(r, f')$, then $f(p) = f'(p)$, which means that f and f' have the same decision for p. □

Theorem 3 (Rule Modification Theorem). *Let f be the given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we modify rule r_i to be r_i' where $1 \le i \le n - 1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_{i-1}, r_i', r_{i+1}, \cdots, r_n \rangle$. We use g to denote the firewall $\langle r_{i+1}, \cdots, r_n \rangle$, which consists of the $n - i$ rules $r_{i+1}, \cdots, r_n$ after rule r_i in firewall f. For any packet p in Σ, consider the following four cases:*

1. if $p \in R(r_i, f) \cap R(r_i', f')$, then $f(p) = r_i.D$ and $f'(p) = r_i'.D$;
2. if $p \in R(r_i, f) - R(r_i', f')$, then $f(p) = r_i.D$ and $f'(p) = g(p)$;
3. if $p \in R(r_i', f') - R(r_i, f)$, then $f(p) = g(p)$ and $f'(p) = r_i'.D$;
4. if $p \in \Sigma - R(r_i, f) \cup R(r_i', f')$, then $f(p) = f'(p)$, which means that f and f' have the same decision for p. □

Theorem 4 (Rule Swap Theorem). *Let f be the given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we swap the two rules r_i and r_j where $1 \le i < j \le n - 1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_{i-1}, r_j, r_{i+1}, \cdots, r_{j-1}, r_i, r_{j+1}, \cdots, r_n \rangle$. Let g be the firewall $\langle r_{i+1}, \cdots, r_n \rangle$, which consists of the $n - i$ rules after rule r_i in firewall f; and g' be the firewall $\langle r_{i+1}, \cdots, r_{j-1}, r_i, r_{j+1}, \cdots, r_n \rangle$, which consists of the $n - i$ rules after rule r_j in firewall f'. For any packet p in Σ, consider the following four cases:*

1. if $p \in R(r_i, f) \cap R(r_j, f')$, then $f(p) = r_i.D$ and $f'(p) = r_j.D$;
2. if $p \in R(r_i, f) - R(r_j, f')$, then $f(p) = r_i.D$ and $f'(p) = g'(p)$;
3. if $p \in R(r_j, f') - R(r_i, f)$, then $f(p) = g(p)$ and $f'(p) = r_j.D$;
4. if $p \in \Sigma - R(r_i, f) \cup R(r_j, f')$, then $f(p) = f'(p)$. □

4   Change-Impact Analysis: Algorithms

In this section, we present algorithms for computing the impact of firewall changes based on the theorems in Section 3. Given a firewall and a proposed change, our change-impact analysis algorithms output a set of so-called "impacts". An impact is of the form

$$\langle predicate \rangle \rightarrow \langle old\ decision \rangle\ vs.\ \langle new\ decision \rangle$$

The meaning of an impact is: the decision of the packets that satisfy the predicate is changed from ⟨*old decision*⟩ to ⟨*new decision*⟩. For ease of understanding, the predicates of all impacts computed for a change are non-overlapping.

4.1 Rule Deletion

Based on Theorem 1, to compute the impacts of deleting rule r_i, we first need to compute the resolving set $R(r_i, f)$. We represent the resolving set of a rule by a set of non-overlapping rules, the union of whose matching sets is exactly the resolving set. This set of non-overlapping rules is called an effective rule set of that rule [3]. More precisely, let r be a rule in a firewall f. A set of non-overlapping rules $\{e_1, e_2, \cdots, e_k\}$ is an *effective rule set* of r iff the following three conditions hold: (1) $R(r, f) = \bigcup_{i=1}^{k} M(e_i)$, (2) $M(r_i') \cap M(r_j') = \emptyset$ for $1 \le i < j \le k$, (3) every e_i has the same decision as r.

How to compute the effective rule set for a rule in a firewall has been discussed in our previous work on removing redundant rules in firewalls [3]. Interested readers can refer to [3] for more technical details. Here we show one example. Considering the example firewall in Figure 1. In this firewall, for simplicity, we assume each packet has only two fields, F_1 and F_2, and the domain of each field is [1,100]. The effective rule set of each rule is shown in Figure 2. Note that we use E_i to denote the effective rule set of rule r_i.

Next, we discuss how to compute the impact of rule deletion through an example. Consider the firewall in Figure 1. Suppose the change is to delete rule r_1, whose effective rule set E_1 is $\{F_1 \in [20, 50] \wedge F_2 \in [1, 70] \rightarrow accept\}$. According to Theorem 1, the question that we need to answer is: which packets that satisfy $F_1 \in [20, 50] \wedge F_2 \in [1, 70]$ are discarded by firewall $\langle r_2, r_3 \rangle$?

To answer this question, we first convert firewall $\langle r_2, r_3 \rangle$ to an equivalent non-overlapping firewall, as shown in Figure 3. (Two firewalls f_1 and f_2 are *equivalent* if and only if for any packet p, we have $f_1(p) = f_2(p)$. A *non-overlapping* firewall is a firewall whose rules are non-overlapping.)

Now the question is: which packets that satisfy $F_1 \in [20, 50] \wedge F_2 \in [1, 70]$ are discarded by the firewall in Figure 3? Given that this firewall is non-overlapping, we can answer this question by checking which discard rule in this firewall overlaps with the predicate $F_1 \in [20, 50] \wedge F_2 \in [1, 70]$. Obviously, the answer is rule r_1'. For the packets that satisfy both predicates $F_1 \in [20, 50] \wedge F_2 \in [1, 70]$ and $F_1 \in [1, 60] \wedge F_2 \in [40, 100]$, their decision by the original firewall is *accept*, but their decision by the modified firewall is *discard*. Therefore, the impact of deleting rule r_1 from the firewall in Figure 2 is as follows:

$$F_1 \in [20, 50] \wedge F_2 \in [40, 70] \rightarrow accept \ vs. \ discard$$

$$
\begin{aligned}
r_1 &: \ F_1 \in [20, 50] \ \wedge F_2 \in [1, \quad 70] \ \rightarrow accept \\
r_2 &: \ F_1 \in [1, \quad 60] \ \wedge F_2 \in [40, 100] \ \rightarrow discard \\
r_3 &: \ F_1 \in [1, \quad 100] \ \wedge F_2 \in [1, \quad 100] \ \rightarrow accept
\end{aligned}
$$

Fig. 1. A firewall example

$$E_1 :\{ \ F_1 \in [20, \ 50] \ \wedge \ F_2 \in [1, \ \ 70] \ \rightarrow accept$$
$$\}$$
$$E_2 :\{F_1 \in [1, \ \ \ 19] \ \wedge \ F_2 \in [40, \ 100] \rightarrow discard$$
$$F_1 \in [51, \ \ 60] \ \wedge \ F_2 \in [40, \ 100] \rightarrow discard$$
$$F_1 \in [20, \ \ 50] \ \wedge \ F_2 \in [71, \ 100] \rightarrow discard$$
$$\}$$
$$E_3 :\{F_1 \in [1, \ \ \ 19] \ \wedge \ F_2 \in [1, \ \ 39] \ \rightarrow discard$$
$$F_1 \in [51, \ \ 60] \ \wedge \ F_2 \in [1, \ \ 39] \ \rightarrow discard$$
$$F_1 \in [61, \ 100] \wedge F_2 \in [1, \ \ 100] \rightarrow discard$$
$$\}$$

Fig. 2. Effective rule sets

$$r'_1 : \ F_1 \in [1, \ \ \ 60] \ \wedge \ F_2 \in [40, \ 100] \ \rightarrow discard$$
$$r'_2 : \ F_1 \in [1, \ \ \ 60] \ \wedge \ F_2 \in [1, \ \ 39] \ \ \rightarrow accept$$
$$r'_3 : \ F_1 \in [61, \ 100] \ \wedge \ F_2 \in [1, \ \ \ 100] \ \rightarrow accept$$

Fig. 3. A non-overlapping firewall
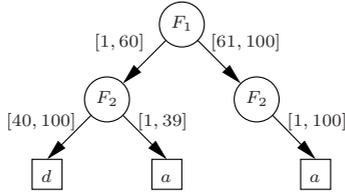
Fig. 4. A firewall decision diagram

For efficiency purposes, we represent a non-overlapping firewall using a firewall decision diagram [4]. For example, the non-overlapping firewall in Figure 3 can be represented using the firewall decision diagram in Figure 4.

The pseudocode of the algorithm for computing the impacts of rule deletion is shown in Figure 5. In this paper, we use *t.root* to denote the root of a firewall decision diagram t, $I(e)$ to denote the label of an edge e, $F(v)$ to denote the label of a node v.

4.2   Rule Insertion

According to Theorem 2, computing the impacts of rule insertion is similar to that of rule deletion. Let f be the given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we insert a new rule r between r_i and r_{i+1} where $1 \leq i \leq n - 1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_i, r, r_{i+1}, \cdots, r_n \rangle$. To compute the impacts of inserting rule r, we first compute the effective rule set of rule r in firewall f'. Second, we construct a firewall decision diagram for the firewall $\langle r_{i+1}, \cdots, r_n \rangle$. (Note that $\langle r_{i+1}, \cdots, r_n \rangle$ is a firewall because the last rule r_n can match any given packet.) Third, we traverse the firewall decision diagram to check which decision path

Computing Impacts of Rule Deletion
Input : A firewall $\langle r_1, \cdots, r_n \rangle$.
Output: Change-impacts of deleting rule r_i.
Steps:
1. Compute the effective rule set E_i of rule r_i;
   Let E_i be $\{e_1, \cdots, e_m\}$.
   $Impacts := \emptyset$;
2. Construct a firewall decision diagram from
   $\langle r_{i+1}, \cdots, r_n \rangle$;
3. **for** $i := 1$ **to** m **do Compare**($t.root$, e_i);
   return E;

Compare(v, $(F_1 \in S_1) \wedge \cdots \wedge (F_d \in S_d) \rightarrow \langle dec \rangle$))
1. **if** (v is a terminal node) and ($\langle dec \rangle \neq F(v)$)
   then/*Let $(F_1 \in S'_1) \wedge \cdots \wedge (F_d \in S'_d) \rightarrow F(v)$
         be the rule defined by the decision path
         containing node v;*/
   $Impacts := Impacts \cup$
   $\{(F_1 \in S_1 \cap S'_1) \wedge \cdots \wedge (F_d \in S_d \cap S'_d) \rightarrow \langle dec \rangle \text{ vs. } F(v)\}$;
2. **if** (v is a nonterminal node) **then**
   /*Let F_j be the label of v*/
   for each edge e in $E(v)$ **do**
     if $I(e) \cap S_j \neq \emptyset$ **then**
       Compare($e.t$, $(F_1 \in S_1) \wedge \cdots \wedge (F_d \in S_d) \rightarrow \langle dec \rangle$))

Fig. 5. Computing impacts of rule deletion

conflicts with a rule in the effective rule set of r. For each conflict discovered, we output an impact. Due to space limitations, we omit the details of the algorithm for computing the impacts of rule insertion.

4.3   Rule Modification

Based on Theorem 3, to compute the impacts of modifying rule r_i in firewall f to be r'_i, we first need to know how to compute $R(r_i, f) \cap R(r'_i, f')$ and $R(r_i, f) - R(r'_i, f')$, where f' denotes the firewall after modifying r_i. Next, we discuss how to compute them.

Given two resolving sets R_a and R_b, which are represented by the effective rule sets $\{e_1, \cdots, e_m\}$ and $\{\varepsilon_1, \cdots, \varepsilon_l\}$ respectively. Then we have $R_a \cap R_b = \cup_{i=1}^{m} \cup_{j=1}^{l} (M(e_i) \cap M(\varepsilon_j))$. Note that $M(e_i) \cap M(\varepsilon_j)$ can be computed as follows. Let rule e_i be $(F_1 \in S_1) \wedge \cdots \wedge (F_d \in S_d) \rightarrow \langle decision \rangle$ and rule ε_j be $(F_1 \in S'_1) \wedge \cdots \wedge (F_d \in S'_d) \rightarrow \langle decision \rangle$. Let rule r be $(F_1 \in S_1 \cap S'_1) \wedge \cdots \wedge (F_d \in S_d \cap S'_d) \rightarrow \langle decision \rangle$. Then we have $M(e_i) \cap M(\varepsilon_j) = M(r)$.

Given two resolving sets R_a and R_b, which are represented by the effective rule sets $\{e_1, \cdots, e_m\}$ and $\{\varepsilon_1, \cdots, \varepsilon_l\}$ respectively. Let r be the rule that any packet can match and f be the firewall $\langle \varepsilon_1, \cdots, \varepsilon_l, e_1, \cdots, e_m, r \rangle$. Then we have $R_a - R_b = \cup_{i=1}^{m} R(e_i, f)$. In other words, $R_a - R_b$ is the union of the effective rule set of every e_i in firewall $\langle \varepsilon_1, \cdots, \varepsilon_l, e_1, \cdots, e_m, r \rangle$.

Let f be the given firewall $\langle r_1, \cdots, r_n \rangle$. Suppose we modify rule r_i to be r_i' where $1 \leq i \leq n-1$. Let f' be the resulting firewall $\langle r_1, \cdots, r_{i-1}, r_i', r_{i+1}, \cdots, r_n \rangle$. The change-impacts of modifying rule r_i can be computed in the following steps:

1. Compute the effective rule set of rule r_i in firewall f, and that of rule r_i' in firewall f'.
2. If r_i and r_i' have the same decision, then skip this step. Otherwise, compute $R(r_i, f) \cap R(r_i', f')$. If $R(r_i, f) \cap R(r_i', f') \neq \emptyset$, then generate impacts accordingly. Note that the decision for any packet in $R(r_i, f) \cap R(r_i', f')$ is changed from the decision of r_i to that of r_i'.
3. Compute $R(r_i, f) - R(r_i', f')$ as follows. Let $\{e_1, \cdots, e_m\}$ and $\{\varepsilon_1, \cdots, \varepsilon_l\}$ be the effective rule sets of rule r_i in firewall f and rule r_i' in firewall f' respectively. Compute the effective rule sets of $e_1, \cdots, e_m$ in firewall $\langle \varepsilon_1, \cdots, \varepsilon_l, e_1, \cdots, e_m, r \rangle$ where r is a rule that any packet can match. Let U be the union of these effective rule sets. Then U represents $R(r_i, f) - R(r_i', f')$.
4. Construct a firewall decision diagram from firewall $\langle r_{i+1}, \cdots, r_n \rangle$.
5. Traverse the firewall decision diagram to check which decision path conflicts with a rule in U. Whenever a conflict is found, our tool outputs an impact.
6. Compute $R(r_i', f') - R(r_i, f)$ by computing the effective rule sets of $\varepsilon_1, \cdots, \varepsilon_l$ in firewall $\langle e_1, \cdots, e_m, \varepsilon_1, \cdots, \varepsilon_l, r \rangle$ where r is a rule that any packet can match. Let U' be the union of these effective rule sets. Then U' represents $R(r_i', f') - R(r_i, f)$.
7. Traverse the firewall decision diagram built from firewall $\langle r_{i+1}, \cdots, r_n \rangle$ to check which decision path conflicts with a rule in U'. Whenever a conflict is found, our tool outputs an impact.

4.4   Rule Swap

Based on Theorem 4, computing the impacts of swapping two rules is similar to that of rule modification. Due to space limitation, we omit the details of the algorithm for computing the impacts of rule swap.

5   Discussion

5.1   Prefix and Intervals

Real-life firewalls usually check five packet fields: source IP address, destination IP address, source port number, destination port number, and protocol type. Of these five fields, the first two fields are usually represented using prefix formats, and the last three fields are usually represented using integer intervals. Note that prefix formats and interval formats are interchangeable. For example, IP prefix 192.168.0.0/16 can be converted to the interval from 192.168.0.0 to 192.168.255.255, where an IP address can be regarded as a 32-bit integer. As another example, the interval $[2, 8]$ can be converted to 3 prefixes: $001*, 01*, 1000$.

To compute firewall change-impacts, we first convert the source and destination IP addresses from prefix formats to integer intervals. Note that every prefix

can be converted to only one integer interval. Second, we run the algorithms
described in Section 4 for computing firewall change-impacts. (Note that the
impacts produced by our algorithms are in interval formats.) Third, for each
impact computed, we convert the source and destination IP addresses from in-
tervals to prefixes. Thus, the format of outputs are similar to that of original
firewall rules, which are easy to understand for firewall administrators. (A $w-$bit
integer interval can be converted to at most $2w - 2$ prefixes [5].)

5.2   High Level Impacts

Many companies or organizations have some "hard requirements" for their fire-
walls. A hard requirement can be interpreted as a set of non-overlapping firewall
rules, which we call "hard rules". For example, assuming that the hard require-
ment for the firewall example in Table 1 is that the mail server 192.168.0.1 should
be able to receive email packets from any host, this hard requirement can be in-
terpreted as the following hard rule

Src IP	Dest. IP	Src Port	Dest. Port	Protocol	Action
*	192.168.0.1	*	25	TCP	accept

After firewall change-impacts are computed, we can compare each impact and
the hard requirements of the firewall, and see which requirements are violated due
to the change. For example, comparing the impact in Table 2 and the above hard
rule, we see that this change violates the above hard rule, and henceforth the hard
requirement. Correlating each impact computed with the hard requirements that
the impact violates is helpful for firewall administrators to verify the correctness
of each impact.

5.3   Making Corrections

After the impacts of a change are computed, the firewall administrator needs
to verify that the impacts are indeed intended. If not all impacts are desirable,
one approach for the firewall administrator is to revise the proposed change
and compute impacts again; another approach for the firewall administrator is
to commit desired impacts by correcting undesired impacts. Next, we show an
example to illustrate the latter approach.

Consider the two impacts in Table 4. If the first impact is exactly what the
administrator intends to do, and the second impact is not desired, we can keep
the proposed change and add the following rule derived from the second (unde-
sired) impact to the beginning of the modified firewall:

Src IP	Dest. IP	Src Port	Dest. Port	Protocol	Action
1.2.3.4	192.168.0.1	*	[25,65536]	TCP	discard

5.4   Complexity Analysis

Let n be the number of rules in a firewall, and d be the total number of distinct packet fields that are examined by a firewall. The complexity of our change-impact analysis algorithms is $O(n^d)$. Despite the high worst case complexities, our algorithms are practical for two reasons. First, d is bounded and is typically small. Real-life firewalls typically examine five packet fields: source IP address, destination IP address, source port number, destination port number, and protocol type. Second, the worst cases of our algorithms are extremely unlikely to happen in practice. To trigger the worst case, the rules in a firewall need to be exceedingly overlapping, which does not happen in real-life firewalls according to the statistics on real-life rule sets in [6].

6   Related Work

Numerous studies have been done on analyzing the change-impact of general programs in software engineering and programming language communities (e.g., [7,8]. However, little work has been done on analyzing the change-impact of firewall policies. Firewall policies and general programs are fundamentally different. While accurately and completely computing the impact of software changes is nearly impossible in general, the algorithms presented in this paper can compute the accurate and complete impact of firewall policy changes.

The closest to this work is our previous work on diverse firewall design [9]. In [9], an algorithm that can compute the semantic differences between two firewalls were presented. The algorithm in [9] can be used to compute change-impact of firewalls by comparing a firewall before changes and the firewall after changes. Comparing with the algorithm in [9], the change-impact analysis algorithms in this paper are much more efficient. Although the algorithm in [9] can handle the cases where a firewall administrator makes multiple changes at a time, in real life, a firewall administrator typically makes one change at a time.

Fisler and Krishnamurthi studied change-impact analysis of access control policies in their seminal paper [10]. They proposed a solution using multi-terminal binary decision diagrams to compute the impact of access control policy changes and verify whether an access control policy satisfies a given property. Their work is similar to ours in spirit, however, their solution cannot be applied to firewall policies because the access control policies studied in [10] are quite different from firewall policies. In [10], every attribute-value pair is encoded as one variable in the MTBDD. This is natural for the access control policies studied in [10], but is not feasible for firewall policies because of the explosive number (2^{88}) of attribute-value pairs.

Some interesting work has been done on firewall policy modelling and analysis. However, none of the previous work explored the change-impact analysis of firewall policies. A Lisp-like language was introduced in [11] for specifying a high-level packet filtering policy. In a similar vein, a UML-like language was presented in [12] for specifying global filtering policies. Detecting conflicts among firewall rules was studied in [13,14,15]. In [16,17,18,19,20], a few firewall analysis

tools were presented. In [21], algorithms for detecting firewall policy anomalies in distributed environment were proposed. Some anomalies were defined and techniques for detecting anomalies were presented in [22, 23].

There is previous work on firewall testing [25, 26, 27, 28, 29]. These testing techniques involve injecting packets into a firewall and checking whether the decisions of the firewall concerning the injected packets are correct. There are some tools currently available for network vulnerability testing, such as Satan and Nessus. These vulnerability testing tools scan a private network based on the current publicly known attacks, rather than the requirement specification of a firewall. Although these tools can possibly catch some of the errors that allow illegitimate access to the private network, they cannot find the errors that disable legitimate communication between the private network and the outside Internet.

7   Conclusions

Making changes to firewall policies is a major task that firewall administrators perform; yet, it is also a major source of firewall policies errors. To address this issue, in this paper, we propose a framework for conducting firewall policy change-impact analysis. Our contributions are three-fold. First, we lay the theory foundation for firewall change-impact analysis. Second, we present algorithms for computing firewall policy change-impacts. Third, we present methods for correlating the impact of a firewall policy change and the high level security requirements that the firewall needs to satisfy as well as methods for making corrections if the impact of a change is not desirable. Our algorithms can be practically used in the iterative process of firewall policy design and maintenance.

References

1. Wool, A.: A quantitative study of firewall configuration errors. IEEE Computer 37(6), 62–67 (2004)
2. Oppenheimer, D., Ganapathi, A., Patterson, D.A.: Why do internet services fail, and what can be done about it? In: Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS-03) (March 2003)
3. Liu, A.X., Gouda, M.G.: Complete redundancy detection in firewalls. In: Jajodia, S., Wijesekera, D. (eds.) Data and Applications Security XIX. LNCS, vol. 3654, pp. 196–209. Springer, Heidelberg (2005)
4. Gouda, M.G., Liu, A.X.: Structured firewall design. Computer Networks Journal 51(4), 1106–1120 (2007)
5. Gupta, P., McKeown, N.: Algorithms for packet classification. IEEE Network 15(2), 24–32 (2001)
6. Gupta, P.: Algorithms for Routing Lookups and Packet Classification. PhD thesis, Stanford University (2000)
7. Horwitz, S.: Identifying the semantic and textual differences between two versions of a program. In: Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation, pp. 234–245. ACM Press, New York (1990)

8. Ren, X., Chesley, O.C., Ryder, B.G.: Using a concept lattice of decomposition slices for program understanding and impact analysis. IEEE Transactions on Software Engineering 32(9), 718–732 (2006)

9. Liu, A.X., Gouda, M.G.: Diverse firewall design. In: DSN-04. Proceedings of the International Conference on Dependable Systems and Networks, pp. 595–604 (June 2004)

10. Fisler, K., Krishnamurthi, S., Meyerovich, L., Tschantz, M.: Verification and change impact analysis of access-control policies. In: Inverardi, P., Jazayeri, M. (eds.) ICSE 2005. LNCS, vol. 4309, Springer, Heidelberg (2006)

11. Guttman, J.D.: Filtering postures: Local enforcement for global policies. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 120–129. IEEE Computer Society Press, Los Alamitos (1997)

12. Bartal, Y., Mayer, A.J., Nissim, K., Wool, A.: Firmato: A novel firewall management toolkit. In: Proceeding of the IEEE Symposium on Security and Privacy, pp. 17–31. IEEE Computer Society Press, Los Alamitos (1999)

13. Hari, A., Suri, S., Parulkar, G.M.: Detecting and resolving packet filter conflicts. In: Proceedings of IEEE INFOCOM, pp. 1203–1212. IEEE Computer Society Press, Los Alamitos (2000)

14. Eppstein, D., Muthukrishnan, S.: Internet packet filter management and rectangle geometry. In: Symp. on Discrete Algorithms, pp. 827–835 (2001)

15. Baboescu, F., Varghese, G.: Fast and scalable conflict detection for packet classifiers. In: Proceedings of the 10th IEEE International Conference on Network Protocols, IEEE Computer Society Press, Los Alamitos (2002)

16. Mayer, A., Wool, A., Ziskind, E.: Fang: A firewall analysis engine. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 177–187. IEEE Computer Society Press, Los Alamitos (2000)

17. Wool, A.: Architecting the lumeta firewall analyzer. In: Proceedings of the 10th USENIX Security Symposium, August 2001, pp. 85–97 (2001)

18. Hazelhurst, S., Attar, A., Sinnappan, R.: Algorithms for improving the dependability of firewall and filter rule lists. In: Proceedings of the Workshop on Dependability of IP Applications, Platforms and Networks (2000)

19. Eronen, P., Zitting, J.: An expert system for analyzing firewall rules. In: Proceedings of the 6th Nordic Workshop on Secure IT Systems (NordSec 2001), pp. 100–107 (2001)

20. Liu, A.X., Gouda, M.G., Ma, H.H., Ngu, A.H.: Firewall queries. In: Higashino, T. (ed.) OPODIS 2004. LNCS, vol. 3544, pp. 124–139. Springer, Heidelberg (2005)

21. Garca-Alfaro, J., Cuppens, F., Cuppens, N.: Analysis of policy anomalies on distributed network security setups. In: Gollmann, D., Meier, J., Sabelfeld, A. (eds.) ESORICS 2006. LNCS, vol. 4189, Springer, Heidelberg (2006)

22. Yuan, L., Chen, H., Mai, J., Chuah, C.N., Su, Z., Mohapatra, P.: Fireman: a toolkit for firewall modeling and analysis. In: IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Los Alamitos (May 2006)

23. Al-Shaer, E., Hamed, H.: Discovery of policy anomalies in distributed firewalls. In: IEEE INFOCOM'04, pp. 2605–2616. IEEE Computer Society Press, Los Alamitos (March 2004)

24. CERT: Test the firewall system,
    http://www.cert.org/security-improvement/practices/p060.html

25. Hoffman, D., Prabhakar, D., Strooper, P.: Testing iptables. In: Proceedings of the 2003 conference of IBM Centre for Advanced Studies, pp. 80–91 (2003)

26. Jürjens, J., Wimmel, G.: Specification-based testing of firewalls. In: Bjørner, D., Broy, M., Zamulin, A.V. (eds.) PSI 2001. LNCS, vol. 2244, Springer, Heidelberg (2001)
27. Hoffman, D., Yoo, K.: Blowtorch: a framework for firewall test automation. In: Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering, pp. 96–103. ACM Press, New York (2005)
28. Senn, D., Basin, D., Caronni, G.: Firewall conformance testing. In: Proceedings of the Testcom (Testing of Communicating Systems) (May 2005)
29. Lyu, M.R., Lau, L.K.Y.: Firewall security: Policies, testing and performance evaluation. In: COMPSAC 2000. Proceedings of the 24th International Conference on Computer Systems and Applications, pp. 116–121 (October 2000)