Inferring the Impact of Firewall Policy Changes by Analyzing Spatial Relations between Packet Filters

Yi Yin, R.S.Bhuvaneswaran, Yoshiaki Katayama and Naohisa Takahashi
Nagoya Institute of Technology Gokiso, Showa-ku, Nagoya, 466- 8555, Japan
E-mail: {yinyi, bhuvan, katayama, naohisa}@moss.elcom.nitech.ac.jp

Abstract—Network security can be increased filtering packets at a firewall. Packet filtering examines network packets and decides whether to accept or deny them, and these decisions are made according to policies that are established by the network administrator and implemented by specific filters. An administrator who finds it hard to understand and maintain a policy, will not easily find problems that occur when the filters are changed (added, deleted, or replaced) and will therefore not be certain that the intended policies are implemented correctly and completely.

In this paper, we consider the relations between filters as spatial relations, and show how the impact of firewall policy changes can be determined by analyzing spatial relations between filters. Using these relations reduces the amount of computation required for impact analysis because it eliminates the need to compare all the predicates involved in the filters. Experimental results show that the proposed impact analysis method is suitable for small networks and can be used for policies with large numbers of filters.

I. INTRODUCTION

Firewalls are important integrated elements in our society, and packet filtering in firewalls increases network security and improves network performance. Packets are examined and accepted or denied according to the policies established by the network administrator, but it is sometimes difficult for an administrator to understand and maintain policies, when the filters in those policies are changed. The administrator therefore always has to wonder whether the intended policies are actually being implemented.

To resolve this problem, we propose an impact analysis method that considers the relations between filters as spatial relations. The term **"impact analysis"** is in software engineering for the activity of identifying what needs to be modified to effect a change, or identifying the potential consequences of a change [8]. Here we use it in the sense of identifying the potential conflicts between those filters in a policy that are changed and the rest of the filters in the policy. The impact analysis method we propose can therefore help administrators assure that their intended policies are indeed being implemented.

We previously presented an anomaly detection system comparing two configurations of packet filters[2], but that system did not enough to present the impacts caused by the policy changes. Al-Share and Hamed presented a firewall advisor based on a detailed classification of anomalies according to their effects [5], but that advisor could identify the relations between filters only by comparing all the predicates involved in the filters.

In this paper, we represent a policy as a space, divide that space into many disjoint sub-spaces (called **cells**), and then compute the spatial relationships between filters by using the inclusion relations of the cells sets. The advantage of our work is that using cells reduces the amount of computation required by eliminating the need to compare all the predicates involved in the filters.

The rest of this paper is organized as follows. In section 2, we introduce packet filtering policy. In section 3, we explain how the relations between filters can be represented by spatial relations, and in section 4, we introduce a method for computing the spatial relations between filters. In section 5, we explain how anomalies are classified, how firewall policies are changed by the addition and deletion of filters, and how the impact of these changes can be analyzed. In section 6, we present and discuss some experimental results. In section 7, we discuss related work, and in section 8, we conclude the paper.

II. BACKGROUND

A firewall policy is an ordered filters set that defines the condition-based actions performed on network packets. If there are n filters in a policy "P", it can be represented as: $P=\{f_1,f_2,\ldots\ldots,f_n\}$.

And the i-th filter is represented as a tuple of four elements: $f_i=(\text{priority}_i, \text{rule}_i, \text{action}_i, \text{policy})$

The "priority_i" represents the priority of filter f_i in "P". In many policies, the priority of a filter is according to the order in which it is described in the filter. That is, a filter with a smaller ordering value has a higher priority.

The "rule_i" represents a set of predicates. The header field of a packet that is analyzed to classify packets is called the **key field**. The key fields commonly used are: protocol type, source IP addresses, destination IP addresses, source port number, and destination port number. A **predicate** is either an equation specifying a key field value for a packet or an inequality condition specifying a range of a key field

values for a packet. SrcIP=123, for example, or 1023≤SrcPort≤65535. If there are m predicates in a rule, the "$rule_i$" can be represented as: $rule_i=(q_1,q_2,\ldots\ldots,q_m)$, where each q_j (j=1,2,……,m) represents a predicate. In this paper, we use a uniform form to represent a predicate. If the original predicate is not in this form it should be translated into it. We use the following three elements tuple to represent a predicate:

$q_j=(k, v_1, v_2)$ $v_1 \le k < v_2$ (v_1=minimum, v_2=maximum+1)

where, k is a key field, v_1 is the minimum value of k, and v_2 is the maximum value plus one. For example, if a predicate 1023≤SrcPort≤65535 is given, we will represent it as q=(SrcPort, 1023, 65536), 1023≤SrcPort<65536. The predicate DesPort=1023 can be represented as q=(DesPort, 1023,1024), 1023≤SrcPort<1024.

The "$action_i$" is the action of filter f_i, either "accept" or "deny". The fourth element "policy" is the policy to which the filter "fi" belongs.

The network packet is accepted or denied by a specific filter if the packet header matches all the predicates of that filter. Otherwise, the next filter in the policy is examined and this process is repeated until a matching filter is found. If none is found, the default filter's action is performed. The default filter of a policy is a filter that can match packets when no other filters of the policy can, and it is always at the last of the filters listed in a policy. An example of a typical policy is shown in Fig.1, where f5 is the default filter.

	Protocol	SrcIP	DesIP	SrcPort	DesPort	Action
f_1:	tcp	*	123.4.5.6	>1023	23	Accept
f_2:	tcp	*	123.4.5.*	>1023	25	Accept
f_3:	tcp	129.6.48.254	123.4.5.9	>1023	119	Accept
f_4:	udp	*	123.4.*.*	>1023	123	Accept
f_5:	tcp	*	*	*	*	Deny

Fig.1: Policy example.

III. SPATIAL RELATIONS BETWEEN PACKET FILTERS

A. *Spatial Representation of Packet Filtering Policy*

In this paper, we use computational geometry to consider packets and filters. When the number of key fields of a packet is N, the packet can be considered a point in an N-dimensional space, called the **packet space**, S^p. A predicate in a filter represents a condition that a key of the packet header should satisfy. A filter with N predicates can thus be considered a portion of S^p, and this portion of S^p is called the **filter space**, SF. A policy consists of a set of filters, so, a set of filter spaces is called a **policy space**, SP. For example, a policy $P_E=\{f_1, f_2\}$ comprised the following filters:

$f_1=(1, (1\le x_0<5, 1\le x_1<4),$ accept, $P_E)$

$f_2=(2, (3\le x_0<6, 3\le x_1<6),$ deny, $P_E)$

Since each of the filters has two keys x_0 and x_1, they can be represented as rectangles in a 2-dimensional spatial space as shown in Fig.2, where we can see that the filter space of "f_1" is the space where $1\le x_0<5$ and $1\le x_1<4$ and that the filter space of "f_2" is the space where $3\le x_0<6$ and $3\le x_1<6$.

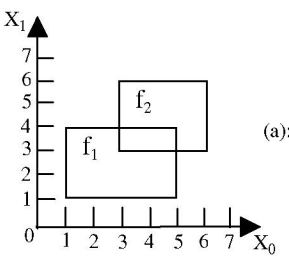

Fig.2: Spatial represent of a policy.

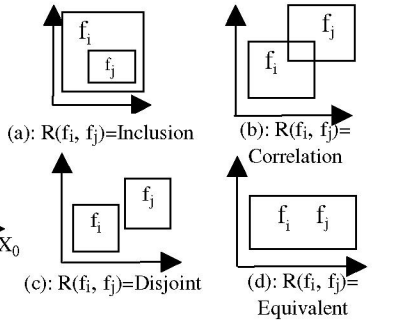

Fig.3: Spatial relations between filters f_i and f_j.

B. *Spatial Relations between Filters*

Spatial relations between any two filters f_i and f_j are classified into the four types. They are represented as $R(f_i, f_j)=r$, where r is an element of {inclusion, correlation, disjoint, equivalent}.

Figure.3(a) shows the $R(f_i, f_j)$=**inclusion** relation between filters f_i and f_j. When the filter space $SF(f_i)$ include the filter space $SF(f_j)$, i.e., when $SF(f_i) \supset SF(f_j)$, a packet that satisfies the condition of filter f_j satisfies the condition of filter "f_i". Conversely, when the filter space $SF(f_i)$ is included by the filter space $SF(f_j)$ when $SF(f_i) \subset SF(f_j)$, a packet satisfying the condition of f_i satisfies the condition of filter f_j.

Figure.3(b) shows the $R(f_i, f_j)$=**correlation** relation between filters f_i and f_j. The filter spaces $SF(f_i)$ and $SF(f_j)$ have a conjunct part. That is, $SF(f_i) \cap SF(f_j) \neq \Phi$. This means that some of the packets that satisfy the condition of f_i also satisfy the condition of filter f_j.

Figure.3(c) shows the $R(f_i, f_j)$=**disjoint** relation between filters f_i and f_j, the filter spaces $SF(f_i)$ and $SF(f_j)$ have no common part. That is, $SF(f_i) \cap SF(f_j) = \Phi$. This means that packets satisfying the condition of filter f_i do not satisfy the condition of filter f_j.

Figure.3(d) shows the $R(f_i, f_j)$=**equivalent** relation between filters f_i and f_j, The filter spaces $SF(f_i)$ and $SF(f_j)$ are the same, or $SF(f_i)=SF(f_j)$. This means that packets satisfy the conditions of filters f_i and f_j at the same time.

IV. IDENTIFY SPATIAL RELATIONS BETWEEN FILTERS

A. *Cells*

To identify the spatial relations between filters, we divide a policy space into disjoint sub-spaces, called **cells**, and represent every filter space by a set of cells. The spatial relations between filters can be decided with inclusion relations of cells sets. The characteristics of a cell are shown as follows:

1) Disjoint: There are no pairs of cells having common sub-space.

2) Direct sum: The union of all cells equals the original policy space.

3) Unitary range: A cell is a sub-space that can be represented by a unitary range value from the first key to the last key.

The cell shape and number of cells depend on the method used to divide the policy space. For example, if we divide the policy space shown in Fig.2 at the minimum and maximum values of the predicates of all the filters in this policy space, we get cells C_0 to C_{16} shown in Fig.4. If we instead let the unitary range as large as possible, we get cells C_0 to C_{10} shown in Fig.5.

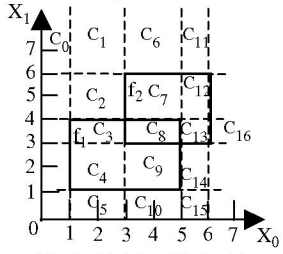

Fig.4: Division Method 1.

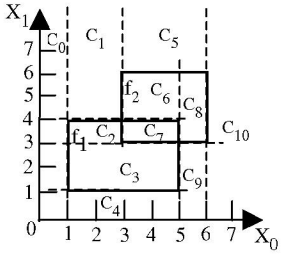

Fig.5: Division Method 2.

B. *Division of Policy Space*

Our division of policy space is based on SIERRA[1], which is a Systolic filter Sieve Array used in a high-speed packet classifier[3]. In section 2, we defined the tuple representing a predicate as $v_1 \leq k < v_2$, (i.e., v_1 is the minimum value of k and v_2 is the maximum value of k plus one. We call v_1 and v_2 the **boundaries** of the predicate. With SIERRA, the whole packet space is first divided into disjoint sub-spaces at all the filter boundaries in the first dimension and then each sub-space is divided at all the filter boundaries in the next dimension. This latter step is repeated until each subspace is divided at all the filter boundaries in the last dimension [1].

The policy space shown in Fig.2, for example, is first divided along the first dimension, the x_0 dimension, into five disjoint sub-spaces S_0 to S_4 as shown in Fig.6. Then each sub-space shown in Fig.6 is divided at the boundaries of all the filters in the x_1 dimension as shown in Fig.7. After the policy space is divided, each disjoint sub-space is one of the cells are shown in Fig.7.

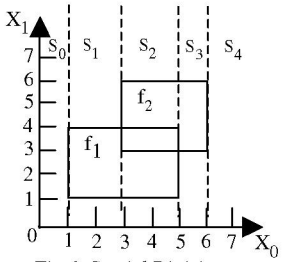

Fig.6: Spatial Division on the x_0 dimension.

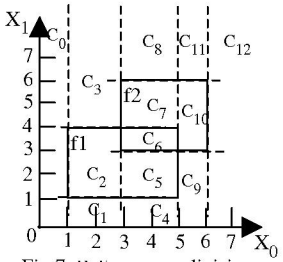

Fig.7: Policy space division.

To represent the relation of filters and cells, we use the two arrays $L[f_i]$ (i=1,2,.....n) and $F[C_j]$ (j=1,2,.....,m).

$L[f_i]$ represents a set of cells that filter f_i covers. For example, in Fig.7, we can see that the filter space of f_1 includes the three cells C_2, C_5, and C_6 and that the filter space of f_2 includes the three cells C_6, C_7, and C_{10}. The arrays $L[f_1]$ and $L[f_2]$ are therefore represented as:

$L[f_1]=\{C_2, C_5, C_6\}$ and $L[f_2]=\{C_6, C_7, C_{10}\}$.

The spatial relations between the filters can be computed by using the inclusion relations of the $L[f_i]$ array. For example, we can say that the spatial relation between filters f_1 and f_2 is a correlation relation, like that shown in Fig.3(b), because the intersection of $\{C_2, C_5, C_6\}$ and $\{C_6, C_7, C_{10}\}$is not empty but contains C_6.

$F[C_j]$ represents a set of filters that the cell C_j belongs to. We call the filters that the cell C_j belongs to **feasible filters** of C_j. In Fig.7, for example, we can see that the cell C_2 belongs to filter f_1 and the cell C_6 belongs to filters f_1 and f_2, the feasible filters of C_2 and C_6 are therefore represented as:

$F[C_2]=\{f_1\}$ and $F[C_6]=\{f_1, f_2\}$.

C. *Cell Representation*

To identify the spatial relations between filters, we use SIERRA tree to represent the cells and their feasible filters.

A SIERRA tree consists of nodes and leaves, and the structure of a node in a SIERRA tree is very similar to the structure of a node in multi-way tree[7]. As shown in Fig.8. each node contains j keys, specified by their boundaries(b_1, b_2, ..., b_j) in one dimension, and j+1 pointers (P_0, P_1, ..., P_j) that point to child nodes. For example, in Fig.6, the boundaries in the x_0 dimension are {1,3,5,6}, so the node corresponding to this dimension division is represented as "node=(P_0, 1, P_1, 3, P_2, 5, P_3, 6, P_4)". "Parent" is the pointer that connects the parent node to the current node. "Depth" is the depth of the current node in the SIERRA tree. "$Child_i$" (i=0,1,2,……,j) is a child node of the current node.

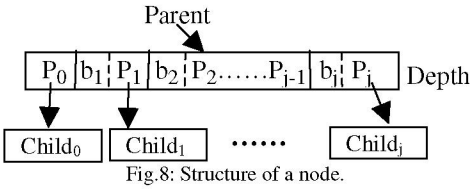

Fig.8: Structure of a node.

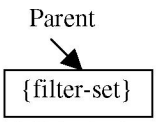

Fig.9: Data structure of leaf.

The structure of a leaf is very simple (Fig.9) and every leaf represents a cell. The filters in a leaf are feasible filters of corresponding cell. The SIERRA tree of policy "P_E" is shown in Fig.10.

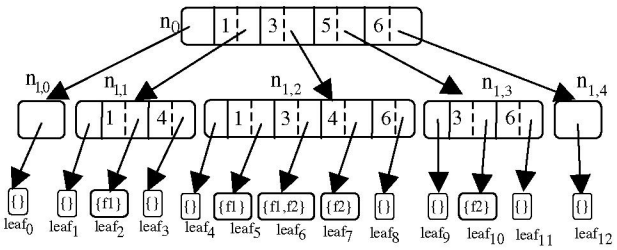

Fig.10: SIERRA tree of P_E.

In a SIERRA tree the root node n_0 represents the division in the x_0 dimension shown in Fig.6, where we can see that the packet space is divided into five sub-spaces S_0-S_4, The root node n_0 in the SIERRA tree therefore has five child nodes. Nodes $n_{1,0}$-$n_{1,4}$ represent the division in the x_1 dimension to each sub-space got from Fig.6. For example,

node $n_{1,1}$ represents the division of subspace S_1 in the x_1 dimension in Fig.6. We can see that in Fig.7, sub-space S_1 is divided into three cells (C_1-C_3) at the boundaries of {1, 4} in the x_1 dimension. Therefore the key values of node $n_{1,1}$ are {1, 4}, and it has three leaves ($leaf_1$ - $leaf_3$).

D. Calculation of Spatial Relations between Filters

Since every leaf in a SIERRA tree represent a cell, filter space also can be represented by a set of leaves, and the spatial relations between filters can be represented by the inclusion relations of sets of leaves.

The array $L[f_i]$ (i=1,2,.....,n) mentioned in subsection 4.2 can be represented by a set of leaves, because a leaf represents a cell. The cells from C_0 to C_{12} in Fig.7 correspond to $leaf_0$ to $leaf_{12}$ in Fig.10, so $L[f_1]$ and $L[f_2]$ can also be represented as:

$L[f_1]$={ $leaf_2$, $leaf_5$, $leaf_6$} and $L[f_2]$={ $leaf_6$, $leaf_7$, $leaf_{10}$}.

The spatial relations between filters f_i and f_j can therefore be represented by inclusion relations between $L[f_i]$ and $L[f_j]$. We suppose that the elements of $L[f_i]$ and $L[f_j]$ are {l_{a1}, l_{a2}, l_{a3},……, l_{ai}} and {l_{b1}, l_{b2}, l_{b3},……, l_{bj}}, where l_{ai} and l_{bj} represent the leaf numbers. The decision method is shown as follows:

$L[f_i]$={ l_{a1}, l_{a2}, l_{a3},……, l_{ai} } (ai=1, 2, 3, ……, n)
$L[f_j]$={ l_{b1}, l_{b2}, l_{b3},……, l_{bj} } (bj=1, 2, 3, ……, n)

$\forall l_{bj} \in L[f_i] \Rightarrow L[f_i] \supset L[f_j] \Rightarrow R(f_i,f_j)$=Inclusion (a)
$\forall l_{ai} \in L[f_j] \Rightarrow L[f_i] \subset L[f_j] \Rightarrow R(f_j,f_i)$=Inclusion (a)
$\exists l_{ai} \in L[f_j]$ and $\exists l_{bj} \in L[f_i] \Rightarrow L[f_i] \cap L[f_j] \neq \varnothing \Rightarrow R(f_i,f_j)$=Correlation (b)
$\nexists l_{ai} \in L[f_j]$ and $\nexists l_{bj} \in L[f_i] \Rightarrow L[f_i] \cap L[f_j] = \varnothing \Rightarrow R(f_i,f_j)$=Disjoint (c)
$\forall l_{bj} \in L[f_i]$ and $\forall l_{ai} \in L[f_j] \Rightarrow L[f_i] = L[f_j] \Rightarrow R(f_i,f_j)$=Equivalent (d)

For example, the spatial relation of filters f_1 and f_2 is determined by the inclusion relation of $L[f_1]$ and $L[f_2]$, and we know that $L[f_1]$ and $L[f_2]$ have a common leaf, $leaf_6$, so $L[f_1] \cap L[f_2] = leaf_6 \neq \varnothing$. We can therefore determine that the spatial relation between f_1 and f_2 is correlation (Fig.3(b)).

We also mentioned the array $F[C_j]$, which represents a set of filters that the cell C_j belongs to. In a SIERRA tree the array $F[leaf_k]$ (k=1,2,.....,n) represents feasible filters of $leaf_k$. For example:

$F[leaf_2]=\{f_1\}$ $F[leaf_6]=\{f_1, f_2\}$ and $F[leaf_7]=\{f_2\}$

V. Analysis of Firewall Policies

Anomalies occur when two or more of the filtering rules used in a firewall match the same packet. In this paper, analyzing the **impact** of firewall changes means identifying the filters affected when the filters in a policy are changed (added, deleted, or modified). Al-Share and Hamed classified possible anomalies in detail according to the relations between the anomalous rule [5], and we used their classification system in the work reported here. In this section, we first introduce the classification of anomalies, then explain how firewall policies change when filters are changed, and finally present our method for analyzing impact of a policy changes by evaluating the spatial relations between filters.

A. Classification of Anomalies in Firewall Filters Policies

When there are two filters f_i and f_j in one policy and f_i precedes f_j, and therefore has a higher priority than f_j. Anomalies and corresponding condition of the spatial relations between f_i and f_j are described as follows:

1. **Shadowing anomaly**: A filter is shadowed if all the packets that match its rules also match the rules of a previous filter. The filter f_j is shadowed by f_i if:
   (1). $R(f_i,f_j)$=Equivalent, f_i.action$\neq f_j$.action
   (2). $R(f_i,f_j)$=Inclusion , f_i.action$\neq f_j$.action

Shadowing is a critical error in the policy, as the shadowed filter never takes effect. This might cause an accepted packet to be denied or a denied packet to be accepted.

2. **Correlation anomaly**: Two filters are correlated if they have different actions, and the first filter matches some packets that match the second filter and vice versa. Filters f_i and f_j have a correlation anomaly if:
   $R(f_i,f_j)$=Correlation, and f_i.action$\neq f_j$.action.

Correlation is considered an anomaly warning because the correlated filters imply an action that is not explicitly stated by the filters.

3. Generalization anomaly: A filter is a generalization of a preceding filter if the two filters have different actions and the first filter can match all the packets that match the second filter. Filter f_j is a generalization filter of f_i if:
   $R(f_j,f_i)$=Inclusion , and f_i.action$\neq f_j$.action.

It is considered only an anomaly warning because the specific filter makes an exception of the general filter. This might cause an accepted packet to be denied or a denied packet to be accepted.

4. **Redundancy anomaly**: A redundant filter performs the same action on the same packets as another filter such that if the redundant filter is removed, the policy will not be affected.
   (1). "f_j" is redundant to "f_i" if:
      ①. $R(f_i,f_j)$=Equivalent, f_i.action = f_j.action
      ②. $R(f_i,f_j)$=Inclusion, f_i.action = f_j.action
   (2). "f_i" is redundant to "f_j" if:

$R(f_j,f_i)$=Inclusion, f_i.action = f_j.action and $\nexists f_k$ where f_i. $priority_i < f_k$. $priority_k < f_j$. $priority_j$, $R(f_i,f_k)$=Inclusion or $R(f_i,f_k)$=Correlation, f_i.action$\neq f_k$.action

Redundancy is considered an error in the policy because a redundant filter unnecessarily increases the size of the filters list, and therefore increases the search time and space requirements of the packet filtering process.

B. Impact of Filter Changes on Firewall Policies

A firewall policy is generally changed when filters are added, deleted, or otherwise modified. We represent filter changes and their impacts on firewall policies as follows:

Change 1) Adding a filter: Administrators often add filters to a policy. Because the semantics of a policy depends on the ordering of filters in the policy, a new filter should be added in the appropriate order.

Suppose, for example, that the following filter f_{new} will be added to the policy shown in Fig.1:

	Protocol	SrcIP	DesIP	SrcPort	DesPort	Action
f_{new}:	tcp	*	123.4.5.7	≥1023	25	Deny

we can see that if f_{new} is added after f_2, it will never take effect, because f_2 will match all the packets that would match f_{new}. The packets that would be denied by f_{new} are accepted by f_2. Unless, f_{new} is added in a position before f_2, it will be shadowed by f_2. Thus a shadowing anomaly is a possible impact of adding f_{new}.

Change 2) Deleting a filter: Deleting a filter affects a policy less than adding a filter does. It will not cause an anomaly but may change the semantics of a policy.

For example, suppose that f_2 is deleted from the policy shown in Fig.11 and we know that packets that would be accepted by f_2 are denied by f_3. The network administrator should then be informed that the semantics of the policy have changed.

	Protocol	SrcIP	DesIP	SrcPort	DesPort	Action
f_1:	tcp	*	123.4.5.6	>1023	23	Accept
f_2:	tcp	*	123.4.5.7	>1023	25	Accept
f_3:	tcp	*	123.4.5.*	>1023	25	Deny
f_4:	tcp	129.6.48.254	123.4.5.9	>1023	119	Accept
f_5:	udp	*	123.4.*.*	>1023	123	Accept
f_6:	tcp	*	*	*	*	Deny

Fig.11: A policy

Change 3) Replacing a filter: Replacing a filter in a policy can be considered the deletion of one filter and the addition of another. For example, suppose that the administrator would like to replace the filter f_1 shown in Fig.1 with the following filter:

	Protocol	SrcIP	DesIP	SrcPort	DesPort	Action
f_{new}:	tcp	*	123.4.5.7	>1023	23	Accept.

The replace operation comprises the deletion of filter f_1 and the addition of filter f_{new}.

C. Impact Analysis

The impact of adding a new filter is analyzed in the following steps:

Step1: Make a SIERRA tree of the current policy (the policy including the added filters and all the filters in old policy).

Step2: To all the leaves in the SIERRA tree, make F[$leaf_k$] (k=1,2,.....,n).

Step3: To all the filters in the policy, make L[f_i] (i=1,2,.....,n)

Step4: In all the leaves in SIERRA tree, check the arrays F[$leaf_k$] (k=1,2,.....,n) to find leaves whose feasible filters include not only the added filter f_{new} but also the other filters f_{other}.

Step5: Take the feasible filters out of each found leaf and use the inclusion relation between L[f_{new}] and L[f_{other}] to identify the spatial relations between the added filter f_{new} and f_{other}.

Step6: Determine the possible impacts that mentioned in section A according to the spatial relations, actions and priorities of the filters f_{new} and f_{other}.

Step7: After determining the possible impacts, decide the priority of f_{new} as follows:

Step7.1: If f_{new} is shadowed by f_{other}, it should be placed before f_{other}, and f_{new}.priority=($f_{(other-1)}$.priority+ f_{other}.priority)/2. If f_{other} is shadowed by f_{new}, f_{new} should be placed after f_{other}, and f_{new}.priority=(f_{other}.priority+ $f_{(other+1)}$.priority)/2.

Step7.2: If f_{new} is correlated with f_{other}, the administrator is notified and decides the priority of f_{new}.

Step7.3: If f_{new} is a generalization of f_{other}, the administrator is notified, and the priority of f_{new} is set equal to (f_{other}.priority+ $f_{(other+1)}$.priority)/2.

Step7.4: If f_{new} is redundant to f_{other}, there is no need to add it; if f_{other} is redundant to f_{new}, f_{other} is deleted, and f_{new}.priority is set equal to f_{other}.priority.

The impact of deleting a filter is analyzed as follows:

Step1 to **Step3** are the same as "Adding a new filter".

Step4 and **Step5** are almost the same as "Adding a new filter" except that the filters of interest are not f_{new} and f_{other} but the deleted filter f_d and f_{other}.

Step6: Inform the administrator of the spatial relations, actions and priorities of the filters f_d and f_{other}, and let the administrator examine the semantics changes. The administrator will decide whether or not the intended policy is still being implemented.

VI. Experiment and Consideration

We evaluated the feasibility and usability of our proposed impact analysis method by measuring the synthesis time and memory usage required by a prototype system. The synthesis time is the time needed for making the SIERRA tree used for policy and anomalies decisions, and the memory usage is the amount of memory needed for making the SIERRA tree used in the impact analysis.

A. Experimental items and Test Date

We measure the synthesis time and memory usage that were needed to detect anomalies when a filter was added to a policy *P*.

We produced policies with random filters in order to represent realistic policies. The number of filters in a policy P for a small network ranges from 5 to 30. To represent all kinds of policies in practical firewalls, we divided filters into the following three groups:

G1: Filters that have the two predicates: SrcIP, and DesIP.

G2: Filters that have the four predicates: SrcIP, DesIP, SrcPort, and DesPort.

G3: Filters that have the seven predicates: SrcIP, DesIP, SrcPort, DesPort, Protocol, length, and TCP flag.

B. Results

The experiment results are listed in Table 1, from which we can see that the synthesis time and memory usage are acceptable. Detecting anomalies when a filter is added to from 5 to 30 filters took 0.94 to 3.98s and 1.18 to 4.18KB of

memory. Hence, our proposed method is feasible for small network firewalls.

When the number of predicates is fixed, and the number of filters increases, the required synthesis time and memory size will increase, which is called "increase A". When the number of predicates increases, and the number of filters is fixed, the required synthesis time and memory size will increase, which is called "increase B". In Table 1, we found that "increase A" is smaller than "increase B". From this result we can confirm that when the number of predicates is fixed and the filter number increases, the synthesis time and memory size will not increase very quickly. Our proposed method can therefore be used for policies with large numbers of filters.

VII. Related Work

"Firewall Policy Advisor" by Al-Share and Hamed[4][5], provides the classification of anomalies in detail. They compare all the predicates of every possible pair of filters in two policies and at the same time make a policy tree. The anomalies between filters are detected after the policy tree is completed.

We use their anomaly classification in the present work, but we represent a policy as a space, and divide the space into disjoint cells. We use a SIERRA tree to represent cells, we identify the spatial relations between filters according to the inclusion relations of set of leaves, and we classify anomalies according to the spatial relations between filters. We used SIERRA tree in our impact analysis method. Most of the computations in making SIERRA tree are included in that of pre-computation of spatial division packet classifier [1][3]. Based on the pre-computation of high-speed packet classifier we can see that our impact analysis method enables to reduce the amount of computation.

Hazelhurst describes a method for transforming a firewall filter specified in a Cisco-like access list language into a BDD (Binary Decision Diagrams) [6]. Although that method can be used to answer questions about the types of packets allowed or excluded by a set of firewall rules and even to find the redundant rules in two policies, it cannot be used to find and classify anomalies. The method we have developed, on the other hand, can not only find anomalies and classify them but also evaluate their impact on packet filtering policies. Although we used a prototype system to evaluate the feasibility and usability of our method, we have efficiently implemented it in software.

VIII. Conclusion

In this paper, we represented the relations between filters spatially and showed how the spatial relations between filters can be used to evaluate the ways in which firewall packet filtering policies are affected when the filters are changed.

The method we proposed is based on four spatial relations between filters and reduces the amount of computation needed for impact analysis because it does not compare all the predicates involved in the filters. The spatial relations between filters are easily inferred form the inclusion relations of sets of leaves sets in a SIERRA tree.

Experimental results show that although the synthesis time and memory size will increase when the number of filters and predicates increases, the proposed method is suitable for small networks and policies with large numbers of filters. Our future plans are to evaluate our impact analysis method in realistic environments to compare it with other methods.

Acknowledgment

This research was partially supported by the Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for JSPS Fellows 1604285, Scientific Research on Priority Areas 18049038 and Scientific Research (C) 18500050, and the Hori Information Science Promotion Foundation.

References

[1] Yi Yin, R.S.Bhuvaneswaran, Y. Katayama, N. Takahashi: " Detection of Anomalies in Packet Filter Configurations", IPSJ SIG Technical Report. Vol.2006, No.42, pp.13-18, 2006, ISSN 0919-6072."

[2] Yi Yin, R.S.Bhuvaneswaran, Y. Katayama, N. Takahashi: "Implementation of Packet Filter Configurations anomaly Detection System with SIERRA", Proc. of 7th International Conference on Information and communications Security (ICICS2005), LNCS Vol.3783, pp.467-480, (2005.12).

[3] N.Takahashi, "A Systolic Sieve Array for Real-time Packet Classification," IPSJ Journal, Vol.42, No.2, pp.146-166(2001)

[4] E.Al-Shaer and H.Hamed. "Discovery of Policy Anomalies in Distributed Firewalls". In Proceedings of IEEE INFOCOM'2004, March 2004.

[5] E.Al-Shaer and H.Hamed. "Firewall Policy Advisor for Anomaly Detection and Rule Editing" IEEE/IFIP Integrated Management Conference (IM'2003), March 2003.

[6] Soctt Hazelhurst, Anton Fatti,and Andrew Henwood. Binary decision diagram representations of firewall and router access lists. Technical Report TR-Wits-CS-1998-3, Department of Computer Science, University of the Witwatersrad, South Africa October 1998

[7] DONALD E.KNUTH, "Art of Computer Programming, Volume 3: Sorting and Searching(2nd Edition)", Addison-Wesley Professional, Boston,Mass.,1998.

[8] Robert S.Arnold, Shawn A.Bohner. "Impact Analysis-Towards A Framework for Comparison". Proceedings of the Conference on Software maintenance pp.292-301, 1993, ISBN: 0-8186-4600-4

Table1: synthesis time and Memory size Needed for Impacts Analysis

Num	**2(G1)**				**4(G2)**				**7(G3)**			
Filters in old P	5	12	20	30	5	12	20	30	5	12	20	30
Add a filter in old P	6	13	21	31	6	13	21	31	6	13	21	31
Time (sec)	0.94	1.31	2.00	2.90	1.62	2.70	3.15	3.50	2.17	3.09	3.42	3.98
Size(kb)	1.18	1.81	3.10	3.67	2.23	3.18	3.78	4.02	2.31	2.93	3.56	4.18

Num: The number of predicates in a filter Time(sec): Synthesis time
Size(KB): Memory usage