Taxonomy of Conflicts in Network Security Policies

Hazem Hamed and Ehab Al-Shaer
School of Computer Science, Telecommunications and Information Systems
DePaul University, Chicago, USA

***Abstract*— Network security polices are essential elements in Internet security devices that provide traffic filtering, integrity, confidentiality and authentication. Network security perimeter devices such as firewalls, IPSec and IDS/IPS devices operate based on locally configured policies. However, configuring network security policies remains a complex and error-prone task due to the rule-dependency semantics, and the interaction between policies in the network. This complexity is likely to increase as the network size increases. A successful deployment of a network security system requires global analysis of policy configurations of all network security devices in order to avoid policy conflicts and inconsistency. Policy conflicts may cause serious security breaches and network vulnerability such as blocking legitimate traffic, permitting unwanted traffic, and insecure data transmission.**

This paper presents a comprehensive classification of security policy conflicts that might potentially exist in a single security device (intra-policy conflicts) or between different network devices (inter-policy conflicts) in enterprise networks. We also show the high probability of creating such conflicts even by expert system administrators and network practitioners.

I. Introduction

With the global connectivity provided by the Internet, network security has gained significant attention in research and industrial communities. Due to the increasing threat of network attacks, network security devices like firewalls and IPSec gateways have become important integrated elements not only in enterprise networks but also in small-size and home networks. On one hand, firewalls provide the network frontier defense against attacks and unauthorized traffic by filtering out unwanted network traffic coming from or going to the secured network. On the other hand, IPSec devices provides integrity, confidentiality and authentication of data communication over IP networks.

Deploying firewall and IPSec technology in the network provides incredible flexibility for customizing the proper protection for different networks and applications. However, our study shows that even expert administrators can make serious mistakes when configuring the network security policy of these devices. Therefore, unawareness of policy conflict types and the lack of automated verification of security policies significantly increase the potential of policy inconsistency and conflicts thus increasing network vulnerability.

There are many challenges confronting the correctness and consistency of security policy configuration in enterprise networks. First, in a single security device, the ordering of the policy rules is critically important to determine the underlying policy semantics. An incorrect rule ordering may obsolete some critical rules and result in a security hole that can be a target for a network attack. Second, the interaction between different network security policies in a distributed network environment introduces additional challenges. For example, inconsistent rule matching between two firewalls can result in illegitimate traffic being allowed into the network, leading to serious security threats such as denial of service attacks. Also, incorrect configuration of cascaded IPSec tunnels can lead to sending confidential data as clear text violating the security policy. Third, the existence of many action types (*e.g.*, bypass, discard, encrypt/tunnel, authenticate/transport, etc.) presents another challenge when analyzing network security policies. Policy conflicts in IPSec devices may cause either a failure in establishing secure communication, or degrade the performance due to performing unnecessary redundant security operations.

Policy conflicts occur due to rule misconfiguration within a single policy (*intra-policy conflicts*) or between policies in different devices (*inter-policy conflicts*). The successful deployment of network security is dependent on thorough understanding of the sources of these conflicts, and enabling automated inspection of security policy rules with minimal human intervention. The goal of this article is to provide a comprehensive road-map for researchers as well as practitioners in the field to study various network security policy conflicts and misconfigurations in firewalls and IPSec gateways. Our conflict classification includes both access-lists and map-lists used respectively by firewalls and IPSec gateways, for both intra- and inter-policy cases. We also identify the conditions and the risk of each conflict or misconfiguration, and show many real examples for illustration. Finally, we briefly describe our implementation of these techniques in a tool called the "Security Policy Advisor" for automatic discovery of security policy conflicts.

Although firewalls and IPSec devices have been deployed for many years, most of the related research has been focused on addressing firewall and IPSec implementation and performance problems. Some related works [4], [7], [11] use a query-based approach to analyze firewall policies. Other related work [5] discovers the conflicts of overlapping IPSec tunnels using a simulation-based technique. To the best of our knowledge, this is the first study that provides a comprehensive classification and identification of rule conflicts in network security devices.

The rest of this paper is organized as follows. In Section II we highlight the main components of network security policies. In Section III we present our formalization of filtering rule relations. In Sections IV and V we identify and define access-list and map-list conflicts, respectively. In Section VI we present an evaluation and usability study of our proposed conflict analysis. In Section VII we give a summary of the previous work related to our research. Finally, in Section VIII we present our conclusions and plans for future research.

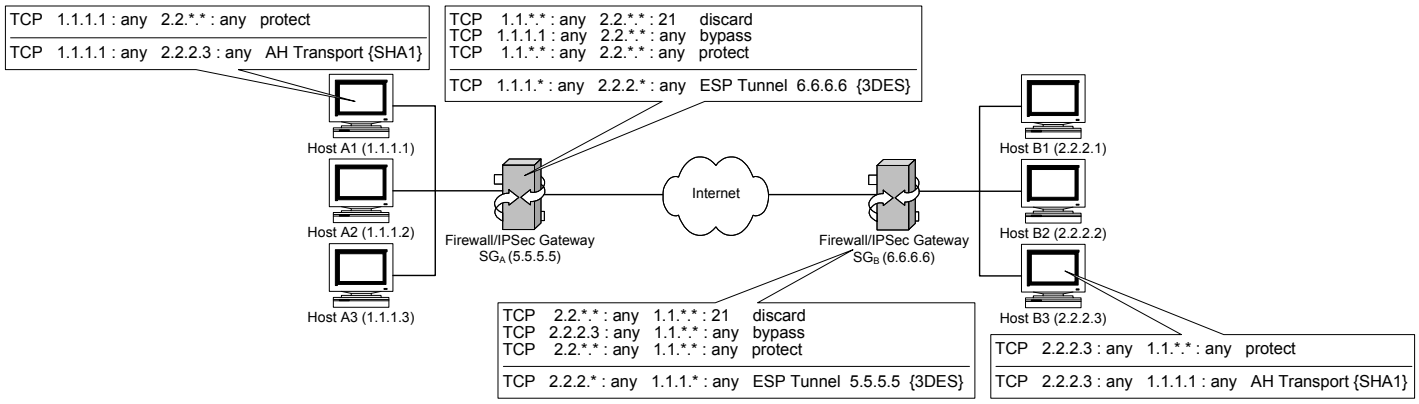

Fig. 1. Example of a typical Firewall/IPSec configuration. Only hosts $A1$ and $B3$ are IPSec capable. Security gateways SG_A and SG_B perform both firewall filtering and IPSec protection operations.

II. Network Security Policy Background

A *security policy enforcement point* is the network element that controls the traversal of packets across network segments based on a given *network security policy*. These include firewalls and IPSec devices, which are installed either at end hosts or at intermediate network nodes. The network protection offered by firewalls/IPSec is based on requirements defined by a security policy established and maintained by a user or system administrator [8]. In general, packets are selected for a packet transmission/protection mode based on network and transport layer header information matched against entries in the policy, *i.e.*, transport protocol, source address and port number, and destination address and port number. Each packet is either discarded, protected and transmitted, or permitted to flow without protection. The decision is based on the policy rules that match this traffic. To define traffic protection rules, we use a generic policy format that resembles the format used in a wide range of firewall and IPSec implementations [12]. In this format, the network security policy is composed of two lists of packet-filtering rules: access list, and map list.

Access list: it consists of ordered filtering rules that define the actions performed on packets that satisfy the rule conditions. All traffic is matched against the access rules sequentially till a matching rule is found. The rules are ordered such that all the packets in a flow will always trigger the same rule. The rule determines the required security action, where a *protect* action means that the traffic is securely transmitted, a *bypass* action means that the traffic is transmitted unsecured and a *discard* action means that the traffic is dropped. Such rules are used in firewall policies to permit or discard traffic, and in IPSec policies to select the traffic to be protected.

Map list: the rules in this list determine the security mapping required to protect the traffic selected by the access list, also based on a set of packet filters. Each rule is given a priority and the traffic is matched against the highest priority rule first. Multiple rules can be triggered for the same flow resulting in applying more than one security transformation on the same traffic. Such rules are used in IPSec policies to specify the cryptographic transformation for a specific traffic.

$$
\begin{array}{rcl}
access\_list & := & access\_rule[\ldots, access\_rule] \\
access\_rule & := & order, filter, action \\
filter & := & transport, src\_ip, src\_port, dst\_ip, dst\_port \\
action & := & \text{protect} \mid \text{bypass} \mid \text{discard} \\
map\_list & := & map\_rule[\ldots, map\_rule] \\
map\_rule & := & priority, filter, transform\_list \\
transform\_list & := & transform[\ldots, transform] \\
transform & := & sec\_protocol, encaps\_mode, parameters \\
sec\_protocol & := & \text{AH} \mid \text{ESP} \\
encaps\_mode & := & \text{Transport} \mid \text{Tunnel } tunnel\_dst
\end{array}
$$

Fig. 2. The syntax of network security policy statements.

Firewalls control the traversal of packets across the boundaries of a secured network based on the security policy. A firewall security policy is an access list of ordered filtering rules. Filtering actions are either *bypass*, which permits the packet to enter or leave the secure network, or *discard*, which causes the packet to be blocked and dropped. Firewalls are widely used to protect the network from external attacks like port and address scanning and denial of service attacks. They are also used to control the flow of traffic between network segments.

IPSec [9] is one of the most widely used protocols to provide secure transmission across the Internet. IPSec uses two security protocols to provide traffic security: Authentication Header (AH) and Encapsulating Security Payload (ESP). The AH protocol provides connectionless integrity, data origin authentication, and an optional anti-replay service. The ESP protocol provides confidentiality (encryption) and optional connectionless integrity, data origin authentication, and an anti-replay service. These protocols may be applied individually or in combination with each other to provide the desired security services. IPSec operations can be performed either at the traffic source and destination (transport mode) or at intermediate *security gateways* (tunnel mode), in order to allow for source-based or domain-based security. In transport mode the protocols provide protection primarily for upper layer protocols without

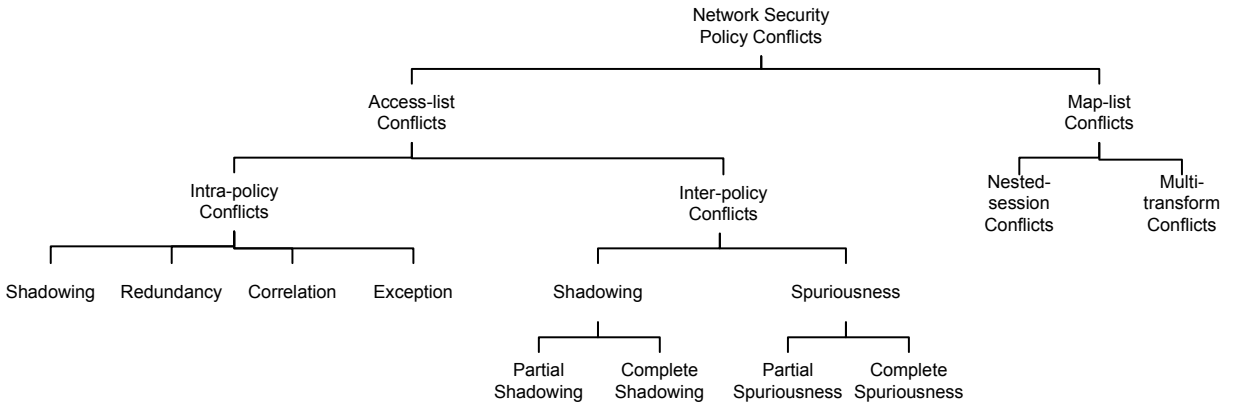

Fig. 3. Classification chart of network security policy conflicts.

changing the original IP headers. However, in tunnel mode, the protocols are applied to the entire IP packet and may modify the source and destination address of the original IP header to be the intermediate security gateways.

Network security policy rules can be written using the syntax shown in Fig. 2. The *access_list* is used to define firewall policies as well as IPSec protection rules, while the *map_list* is used to define IPSec transformation rules. A source or destination IP address can be specified as a single value or a range of values with a common network prefix. A source or destination port number is given as a single value or an interval of values. A *transform* is any cryptographic service that can be used to protect network traffic. These security services are practically IPSec AH and ESP either in transport or tunnel mode along with the cryptographic algorithm and the necessary cryptographic parameters. Fig. 1 shows an example of a typical firewall/IPSec policy for outbound traffic. In this example, security gateways integrate firewall and IPSec functionality. The policy at each device is defined in terms of the access-list (upper section) and the map-list (lower section). In our work, we assume that inbound traffic is matched against a mirror image of the outbound IPSec policy, where the packet filters for the traffic source and destination are swapped [12]. The policy in host $A1$ requires integrity verification using AH transport for all traffic flowing to host $B3$. The policy in gateway SG_A blocks FTP traffic from flowing to any host in network B. In addition, all traffic flowing from network A to network B should be encrypted using ESP tunneling, except for the traffic coming from host $A1$.

In the rest of this paper we proceed with the classification and analysis of conflicts in network security policies. Fig. 3 shows the organization of our taxonomy of these conflicts. We divide the conflicts into two main categories: *access-list* conflicts that may exist between the rules in the access policy, and *map-list* conflicts that may exist in the map policy. We further classify the access-list conflicts into two subcategories: *intra-policy* conflicts that may exist within a single policy, and *inter-policy* conflicts between the policies in different devices. Finally, we identify the possible policy conflicts in each subcategory.

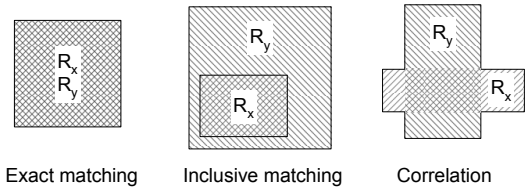

Fig. 4. Different relations between packet filtering rules R_x and R_y.

III. Modeling of Rule Relations

As rules are matched sequentially, the inter-rule relation or dependency is critical for determining any conflict in the security policy. In other words, if the rules are disjoint (no inter-rule relation), then any rule ordering in the security policy is valid. Therefore, classifying all types of possible relations between filtering rules is a first step to understand the source of conflicts due to policy misconfiguration. In this section we describe all the relations that exist between filtering rules by comparing the fields of filtering rules. The relation between field i in rule R_x and the corresponding field in rule R_y can be either equal, subset, superset or distinct. For example, the IP address 140.192.61.5 is distinct from 140.192.60.5, but it is a subset of 140.192.61.*. We can then define the filtering rule relations as follows:

- *Exactly matching rules (R_x=R_y).* Rules R_x and R_y are *exactly matched* if every field in R_x is equal to the corresponding field in R_y. In the following example, Rule 1 and Rule 2 are exactly matching:
  ```
  1: tcp 140.192.37.* any 61.20.33.* 80
  2: tcp 140.192.37.* any 61.20.33.* 80
  ```
- *Inclusively matching rules ($R_x \subset R_y$).* Rule R_x *inclusively matches* R_y if the rules do not exactly match and if every field in R_x is a subset or equal to the corresponding field in R_y. R_x is called the *subset match* while R_y is called the *superset match*. In the following example, Rule 3 inclusively matches Rule 4. Rule 3 is the subset match of the relation while Rule 4 is the superset match:
  ```
  3: tcp 67.92.37.20 any 61.20.33.* 80
  4: tcp 67.92.37.* any 61.20.*.* 80
  ```
- *Correlated rules ($R_x \bowtie R_y$).* Rules R_x and R_y are *correlated* if at least one field in R_x is a subset or partially

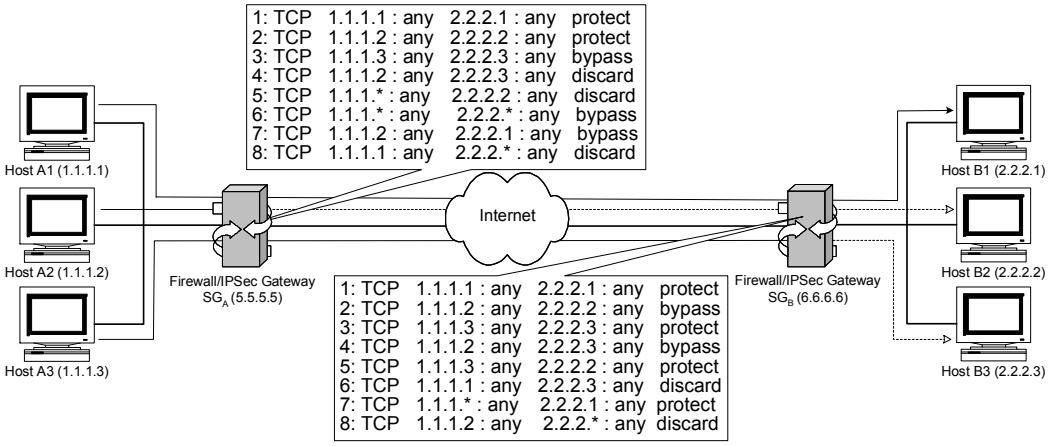

Fig. 5. Example IPSec access-list policy with intra- and inter-policy conflicts. Solid lines indicate permitted traffic, while dotted lines indicate blocked traffic.

intersects with the corresponding field in R_y, and at least one field in R_y is a superset or partially intersects with the corresponding field in R_x, and the rest of the fields are equal. This means that there is an intersection between the address space of the correlated rules although neither one is subset of the other. Figure 4 illustrates the address space intersection in this case. In the following example, Rule 5 and Rule 6 are correlated:

```
5: tcp 140.192.37.* any 61.20.*.* 0-256
6: tcp 140.192.*.* any 61.20.33.* 128-512
```

- *Disjoint rules.* Rules R_x and R_y are *completely disjoint* if every field in R_x is not a subset and not a superset and not equal to the corresponding field in R_y. However, rules R_x and R_y are *partially disjoint* if there is at least one field in R_x that is a subset or a superset or equal to the corresponding field in R_y, and there is at least one field in R_x that is not a subset and not a superset and not equal to the corresponding field in R_y. Rule 7 and Rule 8 in the following example are partially disjoint:

```
7: tcp 140.192.37.* any 61.20.*.* 80
8: tcp 140.192.37.* any 61.20.33.* 21
```

IV. Classification of Access-list Conflicts

After identifying the possible relations between rules, we are now ready to study the types of rule conflicts, In this section, we focus the discussion on the conflicts that may exist between access list rules. These conflicts may exist between rules in the same security device (intra-policy conflicts) or in different devices (inter-policy conflicts).

A. Intra-policy access-list conflicts

It is very common to find filtering rules that are inter-related, *i.e.*, exactly matched, inclusively matched or correlated. In this case, different rule orderings may imply different and incorrect policy semantics (*i.e.*, matching behavior). Thus, if the related rules are not carefully ordered, some rules may be concealed by other rules, resulting in an incorrect policy. In this section, we classify different conflicts that may exist among the rules in a single firewall or IPSec access policy.

Intra-policy shadowing: A rule is shadowed when every packets that could match this rule is matched by some preceding rule with a different action. Consequently, the shadowed rule will never have an effect in the policy. Typically, rule R_y is shadowed by rule R_x if R_x precedes R_y and R_x *exactly* or *inclusively* matches R_y and the two rules have different actions. For example, in Fig. 5 Rule 8 in SG_A is shadowed by Rule 6 because Rule 6 will match and thereby conceal all the traffic that would match Rule 8. Shadowing is a critical error in the policy, as the shadowed rules become obsolete and never take effect. This might cause a legitimate traffic to be blocked or illegitimate traffic to be permitted. Therefore, as a general guideline, if there is an inclusive or exact match relationship between two rules, any superset (or general) rule should come after the subset (or specific) rule. It is mandatory to discover shadowed rules and alert the administrator to correct this error by reordering or removing the shadowed rule.

Intra-policy correlation: As described in Section III, if two rules are correlated then there is some traffic that matches both rules. Thus, if the two correlated rules have different actions, then the action performed on the traffic is dependent on the ordering of the two rules. This dependency is not intuitive and leads to ambiguity in defining the security policy. A correlation conflict between two rules exists if the rules are correlated and they have different filtering actions. For example, a correlation conflict exists between Rule 7 and Rule 8 in SG_B in Fig. 5. The two rules with this ordering imply that all traffic that is coming from 1.1.1.2 and going to 2.2.2.1 is protected. However, if their order is reversed, the same traffic will be discarded. Correlation is considered a conflict warning because the correlated rules imply an action that is not explicitly handled by the policy. Therefore, in order to resolve this conflict, the correlation between rules should be discovered and the user should choose the proper rule order that complies with the security policy requirements. Otherwise, unexpected action might be performed on the traffic that

	$action_x = action_y$		$action_x \neq action_y$	
	$x < y$	$x > y$	$x < y$	$x > y$
$R_x = R_y$	R_y Redund.	R_x Redund.	R_y shadow.	R_x shadow.
$R_x \subset R_y$	R_x Redund.	R_y Redund.	R_x except.	R_y except.
$R_x \bowtie R_y$	Legitimate	Legitimate	R_x correl.	R_y correl.

TABLE I
INTRA-POLICY ACCESS-LIST CONFLICTS.

	$action_u/action_d$		$action_u/action_d$	
	permit/permit	deny/deny	permit/deny	deny/permit
$R_u = R_d$	Legitimate	Legitimate	Comp. spur.	Comp. shad.
$R_u \subset/\bowtie R_d$	Part. shad.	Part. spur.	Comp. spur.	Part. shad.
$R_u \supset R_d$	Part. spur.	Part. shad.	Part. spur.	Comp. shad.

TABLE II
INTER-POLICY ACCESS-LIST CONFLICTS.

matches the intersection of the correlated rules.

Intra-policy exception: Exception rules are very common in network security policies. A rule is an exception of a following rule if they have different actions, and the following rule is a superset match (it could match all packets of the preceding rule). For example, in SG_A in Fig. 5, Rule 2 is an exception of Rule 5. These two rules imply that all the traffic coming from the address 1.1.1.* to 2.2.2.2 will be discarded, except the traffic coming from 1.1.1.2. Exception is often used to exclude a specific part of the traffic from a general filtering action. This is not in general an error, however, it is important to identify exceptions because the exception rules change the policy semantics, which may or may not be desirable. This for example might cause accepted traffic to be blocked or denied traffic to be permitted. Thus, this type of conflict is considered a warning that is important to be highlighted to the administrator.

Intra-policy redundancy: A rule is redundant when every packets that could match this rule is matched by some other rule that has a similar action. In other words, if the redundant rule is removed, the behavior of the security policy remains unchanged. Rule R_y is redundant to rule R_x if R_x precedes R_y and R_y *exactly* or *inclusively* matches R_x and the two rules have similar actions. However, rule R_x can be redundant to R_y if R_x *inclusively* matches R_y and there exists no rule in-between that is an exception or correlated with R_x. Fig. 5 shows two rule redundancy examples in SG_A; Rule 3 with Rule 6 is the first case, and Rule 7 with Rule 6 is the second. Redundancy is considered a problem in firewall policies because it increases the policy size and consequently the search time and space of packet filtering, while it does not contribute to the policy semantics. Thus, it is important to discover redundant rules so that the administrator may modify or remove them altogether. In general, to avoid redundant rules, a superset rule following a subset rule should have a different filtering action.

Table I summarizes the conditions of the intra-policy access-list conflicts. The table shows the type of conflict existing between two rules with different rule relations and actions.

B. Inter-policy access-list conflicts

Even if every device policy in the network does not contain any of the intra-policy conflicts described in Section IV-A, conflicts could exist between policies of different devices. For example, a firewall might block traffic that is permitted by another downstream firewall, or an upstream IPSec device might protect traffic that is not protected by the peer downstream device. In both cases, the traffic will not reach the destination as it will be dropped by the upstream devices. In this section, we first define criteria of inter-policy conflicts and then classify the conflicts that may exist between an upstream and a downstream access policy for both firewall and IPSec devices.

Referring to Fig. 5, we assume a traffic stream flowing from sub-domain D_A to sub-domain D_B across multiple cascaded policy enforcement points installed on the network path between the two sub-domains. At any point on this path in the direction of flow, a preceding device is called an *upstream device* whereas a following one is called a *downstream device*. Using the above network model, we can say that for any traffic flowing from sub-domain D_A to sub-domain D_B a conflict exists if one of the following conditions holds:

1) The most-downstream device permits traffic that is blocked by any of the upstream devices.
2) The most-upstream device permits traffic that is blocked by any of the downstream devices.

On the other hand, all intermediate devices should permit/bypass any traffic that is permitted/protected by the most-upstream and most-downstream device such that the flow can reach its destination.

a) Inter-policy shadowing: This conflict is similar to the one discussed in Section IV-A except that it occurs between rules in two different policies/devices. Thus, Inter-policy shadowing occurs if the upstream policy blocks some traffic that is permitted by the downstream policy.

There are two cases for shadowing between polices. The first case occurs if the traffic permitted (or protected) by a downstream rule, R_d, in SG_B is actually discarded by the upstream rule, R_u, in SG_A. In the second case, the traffic protection is required by R_u in SG_A but not by any R_d in SG_B. In IPSec, this causes the security association negotiation to fail and consequently results in discarding this traffic at the upstream device. In Fig. 5, Rule 4 in SG_A and Rule 4 in SG_B show an example of the first case. An example for the second case is Rule 2 in SG_A and Rule 2 in SG_B. If the rules exactly match, then *all* the traffic permitted by R_d is blocked by R_u, and we have *complete shadowing*. The two examples above are considered complete shadowing. However, if the rules inclusively match, then part of the traffic permitted by R_d is blocked by R_u, and we have *partial shadowing*. Rule 5 in SG_A and Rule 5 in SG_B are an example of partial shadowing.

Shadowing is considered a conflict because it prevents the traffic desired by some nodes from flowing to the end destination.

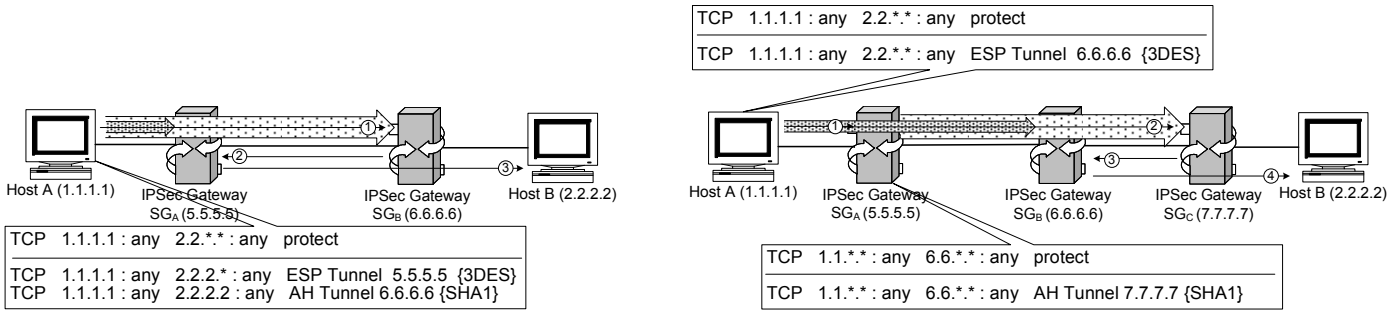

Fig. 6. Examples for overlapping-session conflicts in the policy of (a) single device, and (b) multiple devices. The block arrows indicate the tunnel between two end-points. The line arrows indicate the traffic flow path with the sequence of events circled.

b) Inter-policy spuriousness: This is another conflict that has serious consequence for network security as it allows for unwanted traffic to flow in the network. Traffic is called spurious if the upstream policy permits some traffic that is blocked by a downstream policy.

Spuriousness can occur when traffic is permitted by an upstream rule, R_u, in SG_A but discarded (or protected) by a downstream rule, R_d, in SG_B. Obviously, this traffic will not reach the destination domain D_B but yet it was unnecessarily allowed to flow through the network till it got discarded. In Fig 5, Rule 3 in SG_A and Rule 3 in SG_B show an example of spuriousness. If the rules exactly match, then all the traffic permitted by R_u is blocked by R_d, and we have *complete spuriousness*. If the rules inclusively match, then part of the traffic permitted by R_u is blocked by R_d, and we have *partial spuriousness*. An example of partial spuriousness is Rule 6 in SG_A and Rule 6 in SG_B in the same figure.

Spuriousness is a critical conflict because it allows unwanted traffic to flow across the network, increasing the network vulnerability to various network attacks such as port scanning, denial of service, etc.

Table II summarizes the conditions of the inter-policy access-list conflicts. The table shows the types of conflicts existing between upstream and downstream rules with different rule relations and actions.

V. Classification of Map-list Conflicts

The map-list is critical because it is the part of the policy that specifies the traffic security requirements. In this section, we identify the rule conflicts that may exist between the map list rules in a single (intra-policy) IPSec device or between multiple (inter-policy) IPSec devices. As a result of such conflicts, traffic might be transmitted insecurely or unnecessary redundant protection operations are applied to the same traffic. We also show examples of these conflicts in IPSec traffic transformation policies.

A. Overlapping-session conflicts

IPSec security policy allows for nesting multiple secure sessions or tunnels by applying a number of map-list rules to the same traffic (*e.g.*, Fig 6(a)). In case of intra-policy nested IPSec sessions, each session starts at the source (encapsulation point) and terminates at a selected peer on the path (decapsulation point) specified in the rule, and the sessions are applied in the order of their priorities in the policy. If the decapsulation point of the firstly applied session (inner tunnel) (*e.g.*, 5.5.5.5 in Fig 6(a)) happens to be located before that of a subsequent session (outer tunnel) (*e.g.*, 6.6.6.6 in Fig 6(a)) on the traffic path, then the traffic will be transmitted insecurely as clear text. This happens simply because the destination peer of the outer tunnel in this case returns the traffic after decapsulation back to the destination peer of the inner tunnel, which in turn decapsulates and forwards the traffic to the destination as clear text. This tunnel overlapping conflict occurs because the rules were not ordered correctly in the map-list such that the priorities of IPSec sessions terminating at further points from the source are higher than the priorities of the ones with closer termination points. The example in Fig. 6(a) shows two map rules applying to the traffic flowing from host A to B. The first rule encapsulates the traffic in a tunnel to SG_A, then the second rule re-encapsulates the traffic in another tunnel to SG_B. However, this rule ordering results in incorrect nesting that causes the traffic to be sent back from SG_B to SG_A, and then decapsulated and forwarded as clear text to B.

Similarly, the same problem might happen due to inter-policy conflict between multiple IPSec gateways. However, in this case the only difference is the encapsulation point (start of the tunnel) might be at different point on the path. Fig. 6(b) shows an example of two overlapping tunnels: from host A to 6.6.6.6 and from 5.5.5.5 to 7.7.7.7. As in the previous case, the outer tunnel decapsulation peer (7.7.7.7) comes after the inner tunnel decapsulation peer (6.6.6.6) on the path, causing a similar security violation. The example in Fig. 6(b) illustrates this case. In this example, two IPSec sessions are used to protect the traffic flowing from A to B. The first session starts at A and encapsulates the traffic in a tunnel terminating at SG_B. The second session starts at SG_A and re-encapsulates the traffic in another tunnel terminating at SG_C. The traffic is first received and decapsulated by SG_C and then forwarded back to SG_B. SG_B decapsulated the traffic and forwards it to B as clear text.

In order to assure correct overlapping of inter-policy tunnels, we must guarantee the decapsulation peers of the outer tunnels come first in the path. In other words, the packets should be decapsulated in reverse order of their encapsulation at subsequent points on the traffic path. This requires that the sessions for the same traffic must be either completely nested (*i.e.*, the end-points of one session lie between the end-points of the other

	Intra-Policy Conflicts			Inter-Policy Conflicts		
Experience	Access-list Conflicts	Overlapping-session Conflicts	Multi-transform Conflicts	Access-list Conflicts	Overlapping-session Conflicts	Multi-transform Conflicts
Expert	14%	14%	0%	29%	14%	14%
Intermediate	42%	33%	8%	50%	33%	17%
Beginner	84%	63%	16%	90%	53%	16%
Conflict type %	19%	9%	7%	38%	16%	11%

TABLE III

THE PERCENTAGE OF ADMINISTRATORS WHO CREATED INTRA- AND INTER-POLICY IPSEC CONFLICTS.

session), or cascaded (*i.e.*, none of the end-points of any session lies between the end-points of the other session). Otherwise, the traffic will be sent unsecured over parts of the path due to this conflict.

So in general, by looking at any IPSec policy, this conflict exists if two rules match a common flow, and the tunnel end-point of the firstly applied rule comes before the tunnel end-point of the following rule in the path from source to destination. Notice that this conflict can only occur with two tunneled transforms or with a transport transform followed by a tunnel.

B. Multi-transform conflicts

IPSec also allows for multiple transforms to be applied by the same gateway or multiple gateways in the path on the same flow from source to destination. This gives the user the flexibility to combine different IPSec protection actions on the same traffic. However, some of these combinations provide weak protection, such as applying ESP transport after AH transport because ESP transport does not provide IP header protection. Moreover, other combinations may cause extra overhead without any further improvement in the traffic protection, particularly if the new protection is weaker than the existing one. For example, applying an AH tunnel on traffic already encapsulated in an ESP tunnel does not improve the security protection.

The multi-transform conflict occurs when two rules match a common flow, and the secondly applied rule uses a weaker transform on top of a stronger one applied by the other rule. For flexibility, the strength of any transform can be user-defined such that if a transformation has a larger strength value, then it provides better protection, and vice versa.

VI. IMPLEMENTATION AND EVALUATION

In this section, we evaluate the effectiveness of our classification in discovering the conflicts in manually produced security policies. We implemented a set of conflict discovery algorithms to discover the security policy conflicts studied in Sections IV and V [2]. In these algorithms, we represent the security policy as Boolean expressions in order to to enable the analysis of policy semantics and the identification of rule conflicts using Boolean operators such as assignment and implication. We use Ordered Binary Decision Diagram (OBDD) [3] well-founded methods to represent and manipulate the policy expressions. We implemented these algorithms in a software tool called the "Security Policy Advisor" or SPA. In this section, we present our evaluation of the usability of the policy analysis techniques described in this paper.

To assess the practical value of our techniques, we used the SPA tool to analyze real firewall and IPSec policy rules in our university network as well as in some local industrial networks in the area. In many cases, the SPA has proven to be effective by discovering many policy conflicts that were not discovered through human visual inspection. We made an attempt to quantitatively evaluate the practical usability of the SPA by conducting a set of experiments that consider the level of network administrator expertise and the frequency of occurrence of each conflict type. In this experiment, we created two IPSec policy exercises and recruited 38 network administrators with varying level of expertise in the field (7 experts, 12 intermediate and 19 beginners) to complete each exercise. The exercise included writing IPSec access list and map list rules based on a set of security policy requirements for 9 interconnected networks with 12 IPSec security gateways (intermediate and end-point gateways.) We then used the SPA tool to analyze the rules and count different types of conflicts. The experimental results in Table III show the percentage of individuals who introduced various types of conflicts in their IPSec policy configurations.

These results show clearly that even the expert administrators created policy conflicts. A total of about 29% of experts created intra-policy and inter-policy conflicts. This figure is even much higher for intermediate and beginner administrators (a total of about 67% and 90% created intra-policy conflicts, and 75% and 95% created inter-policy conflicts, respectively). An interesting observation is that most of the persons made configuration mistakes in configuring access-list and overlapping-session rules. The table also shows the average ratio of each conflict type relative to the total number of conflicts discovered for each class of administrators. The results clearly indicate that access-list conflicts dominate the misconfiguration errors made by administrators (19% intra-policy and 38% inter-policy access-list conflicts).

VII. RELATED WORK

A significant amount of work has been reported in the area of firewall and policy-based security management. In this section, we shall focus our study on the research work related to policy analysis and conflict discovery in filtering rules.

A variety of approaches have been proposed in the area of policy conflict analysis. The most significant approach to IPSec policy analysis is proposed in [5]. The technique simulates IPSec processing and reports any violation of the security policy requirements. Although this approach can discover IPSec policy violations in a certain simulation scenario, there is no guarantee that it discovers every possible violation that may exist. In

addition, the proposed technique only discovers IPSec conflicts resulting from incorrect tunnel overlap, but does not address the other types of conflicts that we study in this paper. Our previous work in firewall policy analysis [1] was a significant advance in the area. However, this work was limited to firewall policy analysis where the policy triggers a single filtering action that is limited only to "permit" and "discard". The authors in [6] provide algorithms for detecting and resolving conflicts among general packet filters. However, they only recognize what we defined as intra-policy correlation conflict because it causes ambiguity in packet classifiers. Other research work goes one step further by offering query-based tools for firewall policy analysis [4], [7], [11]. Even though these tools can be extended to run queries to analyze network security policies, they cannot provide pre-defined and automated policy conflict discovery. Moreover, they have limited practical usability as they require significant user expertise to write the proper queries to identify different policy problems. Other work in this area addresses general management policies rather than filtering policies [10]. Although this work is very useful as a general background, it cannot be directly used for conflict discovery in network security policies.

Therefore, based on our search, we could not find any previous work that offers a comprehensive conflict analysis framework for general network security policies.

VIII. Conclusions and Future Work

Although firewalls and IPSec provide flexible traffic control and data protection schemes for IP networks, configuring the security policies of these devices is a critical task, particularly in enterprise networks. Hundreds to thousands of policy rules that may exist in various policy enforcement devices need to be analyzed locally and globally in order to discover intra- and inter-policy conflicts. Improper configuration of these devices can lead to serious network vulnerabilities and threats such as flooding attacks and insecure transmission. Our field study indicates that human error is imminent since almost 30% of expert system administrators made configuration mistakes that lead to serious policy conflicts. Network security, like any other technology, requires proper management support including automatic conflict analysis and verification in order to provide the required security service.

The main contribution of this paper is the comprehensive classification of the conflicts in filtering-based network security policies. These conflicts include improper traffic flow control, like shadowing and spuriousness conflicts, as well as incorrect traffic protection, like conflicts between nested/overlapping security sessions. We also present easy-to-follow guidelines to identify and rectify these conflicts. Our approach is sufficiently general that it can be used to verify many other filtering-based security policies such as authorization servers, intrusion detection, and intrusion prevention systems.

We believe that much more research work is needed in the security policy management area. Our future research plan includes automatic discovery and recovery of conflicts created as a result of policy updates, and high-level network security policy definition and visualization.

References

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

[2] H. Hamed and E. Al-Shaer and W. Marrero. "Modeling and Verification of IPSec and VPN Security Policies." *Proceedings of IEEE ICNP'2005*, November 2005.

[3] R. Bryant. "Graph-Based Algorithms for Boolean Function Manipulation." *IEEE Transactions on Computers*, August 1986.

[4] P. Eronen and J. Zitting. "An Expert System for Analyzing Firewall Rules." *Proceedings of 6^{th} Nordic Workshop on Secure IT-Systems (NordSec 2001)*, November 2001.

[5] Z. Fu, F. Wu, H. Huang, K. Loh, F. Gong, I. Baldine and C. Xu. "IPSec/VPN Security Policy: Correctness, Conflict Detection and Resolution." *Proceedings of Policy'2001 Workshop*, January 2001.

[6] B. Hari, S. Suri and G. Parulkar. "Detecting and Resolving Packet Filter Conflicts." *Proceedings of IEEE INFOCOM'00*, March 2000.

[7] S. Hazelhusrt. "Algorithms for Analyzing Firewall and Router Access Lists." *Technical Report TR-WitsCS-1999*, Department of Computer Science, University of the Witwatersrand, South Africa, July 1999.

[8] K. Jason, L. Rafalow and E. Vyncke. "IPsec Configuration Policy Information Model." *RFC 3585*, IETF, August 2003.

[9] S. Kent and R. Atkinson. "Security Architecture for the Internet Protocol." *RFC-2401*, IETF, November 1998.

[10] E. Lupu and M. Sloman. "Conflict Analysis for Management Policies." *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM'1997)*, May 1997.

[11] A. Mayer, A. Wool and E. Ziskind. "Fang: A Firewall Analysis Engine." *Proceedings of 2000 IEEE Symposium on Security and Privacy*, May 2000.

[12] "Configuring IPSec Network Security." *Cisco IOS Security Configuration Guide, Release 12.2*, Cisco Systems, Inc.