On Assisted Packet Filter Conflicts Resolution: An Iterative Relaxed Approach

Anis Yazidi
Department of Computer Science
Oslo and Akershus University College
Oslo, Norway

Adel Bouhoula
Digital Security Research Unit
Higher School of Communication of Tunis (Sup'Com)
University of Carthage -Tunisia
adel.bouhoula@supcom.tn

***Abstract*—With the dramatic growth of network attacks, a new set of challenges has raised in the field of electronic security. Undoubtedly, firewalls are core elements in the network security architecture. However, firewalls may include policy anomalies resulting in critical network vulnerabilities. A substantial step towards ensuring network security is resolving packet filter conflicts. Numerous studies have investigated the discovery and analysis of filtering rules anomalies. However, no such emphasis was given to the resolution of these anomalies. Legacy work for correcting anomalies operate with the premise of creating totally disjunctive rules. Unfortunately, such solutions are impractical from implementation point of view as they lead to an explosion of the number of firewall rules.**

In this paper, we present a new approach for performing assisted corrective actions, which in contrast to the-state-of-the-art family of radically disjunctive approaches, does not lead to a prohibitive increase of the firewall size. In this sense, we allow *relaxation* in the correction process by clearly distinguishing between constructive anomalies that can be tolerated and destructive anomalies that should be systematically fixed. This distinction between constructive and destructive anomalies is assisted by the network administrator which supports the fact that he has a major role in the heart of the corrective process. To the best of our knowledge, such assisted approach for relaxed resolution of packet filter conflicts was not investigated before. We provide theoretical analysis that demonstrate that our scheme results is sound and indeed result into a conflict-free policy. In addition, we have implemented our solution in a user friendly tool.

***Index Terms*—Firewall Policy, Filtering Rules, Anomalies Discovery, Anomalies Correction**

I. Introduction

With the dramatic growth of the Internet, network security has become a focal concern during this last decade. Firewalls are widely deployed in private networks as an inherent part of their security. However, the effectiveness of a firewall is dramatically jeopardized by the presence of anomalies within its filtering rules. In fact, the filtering rules may include anomalies resulting in critical security vulnerabilities.

The analyses of filtering rules and anomalies discovery have gained a lot of attention. A significant work was reported in this area [1]–[13]. Most of the emphasis has been given to the classification and discovery of firewall anomalies. Other studies have focused on optimizing the filtering process time [14]–[16]. However, few studies have been performed to resolve these anomalies. The most notable of these studies are [1] and [17] which only focused on one of the conflict problems, namely, rule correlation in filtering policies. Other remarkable studies [2], [18] have defined a set of recommendations for correcting policy anomalies. However [18] and [2] did not develop a concrete approach to resolve packet filter conflicts. Another study that tried to probe into the conflict resolution issue is reported in [19]. The main shortcoming of [19] is that the proposed corrective actions do not handle the correlation anomaly. In [20]–[22], a set of algorithms for rewriting firewall rules were presented. The new rewritten rules are completely free of errors and equivalent to the initial misconfigured firewall rules. However, the complexity of the algorithm is very high and leads to an explosion in the number of the firewall rules.

Dealing with an assisted correction of policy anomalies seems to be a captivating and challenging task. In fact the correction is a highly complicated task that threatens to overwhelm human attention as the number of rules increases. Moreover, this task is prone to errors. In fact, modifying the rules order may create new anomalies instead of correcting the existent ones. Hence, a special attention has to be paid when ordering the rules. From this perspective raises the need of an automatic system to correct anomalies within filtering rules.

In this paper, we address the problem of correcting the anomalies within filtering rules. We prove that ordering does not always work to correct the anomalies defined in [18]. In some cases, we will add new rules to achieve the stability of the firewall. In our work, we adopt the formal definitions of all possible filtering rules anomalies stated in [18], [23]. This model is complete (i.e. includes all rules possible in any filtering policy). We make use of this model to classify the filtering policy anomalies. Therefore, our work presents a significant contribution in this field since it offers a new approach to resolve anomalies within filtering rules based on the complete definition of anomalies stated in the work of Al-Shaer and Hamed [18]. We should underline that our corrective system is not fully automatic. In fact, we support the claim state in the study [18] highlighting that the administrator has a major role in the corrective process. Hence, our corrective system is guided by the administrator choices in order to reflect exactly the desired policy. Our paradigm is based on a multi-stage corrective process. In the first stage, we eliminate the anomalies: contradiction and redundancy. In the second stage, we draw precedence relationships from the anomalies

DOI 10.1109/LCN.2016.15

discovery process. In the third stage, we resort to these local precedence relationships in order to achieve a global order of the filtering rules that reflects the semantic policy adopted by the administrator, with the eventuality of adding new rules.

This paper is organized as follows. In section II, we survey the definitions of all possible anomalies. In section III, we describe our scheme to tackle the problem and provide some theoretical results that prove its soundness. In section IV, we present our software implementation and two examples that illustrate our approach. Conclusions are drawn in section V.

II. Firewall Anomaly Classification

For the sake of clarity, it is important to present a background on the subject. In this perspective, we will introduce some useful definitions then we will focus on the types of anomalies.

A. Definitions

The following definitions are essential to follow the paper.

Definition 1: [Header]

A header is a set of variables depending on the protocol. The header is generally characterized by the following list of fields:

Header = <Protocol> <Source IP> <Destination IP> <Source port> <Destination port>

Note that this common header format is not unique and changes may occur depending on the protocol type. In fact, we distinguish various header formats namely UDP, TCP, ICMP and ARP/RARP header formats.

Definition 2: [Filtering Rule]

A filtering rule is the concatenation of header fields and an action. The action defines whether the given packet should be accepted or rejected. In this sense, a filtering rule can be written as :

Filtering Rule = <Header> <Action>

Let R be filtering rule. We denote $R[i]$ the value of the field number i, where $i \in$ {Protocol, Source IP, Destination IP, Source port, Destination port}

Definition 3: [Disjunction]

The two rules R_x and R_y are disjoint if there is at least one pair of fields which are disjoint. Formally, $R_x \neq R_y$ iff $\exists k$ such as $R_x[k] \bigcap R_y[k] = \emptyset$.

A key note to underline is that two disjoint rules are anomaly free.

Definition 4: [Equality Relationship]

The two rules R_x and R_y are said to be equal if every field in R_x is equal to the corresponding field in R_y. Formally,

$R_x = R_y$ iff $\forall$ i, $R_x[i] = R_y[i]$

Types of anomalies:

The inter-relation between filtering rules may result in conflicts in packet classification .To resolve these conflicts it is important to identify them first. We adopt the same definitions of anomalies developed by E. Al-Shaer. For more details we refer the reader to [18]. In addition, we introduce the definition of the contradiction anomaly stated in [2]. The contradiction anomaly is merely a particular case of the shadowing anomaly.

Definition 5: [Contradiction Anomaly]

Two rules present the contradiction anomaly if they are equal and they have different actions. Contradiction is a particular case of shadowing. However, we will clearly differentiate in this paper between shadowing and contradiction.

Definition 6: [Shadowing Anomaly]

A rule is shadowed when a previous rule matches all the packets that match this rule, such that the shadowed rule will never be activated. Besides, the involved rules should not be part of the contradiction anomaly.

Definition 7: [Correlation Anomaly]

Two rules are correlated if they have different filtering actions and first rule matches some packets that match the second rule and the second rule matches some packets that match the first rule.

Definition 8: [Generalization Anomaly]

A rule is a generalization of a preceding rule if they have different actions, and if the first rule can match all the packets that match the second rule.

Definition 9: [Redundancy Anomaly]

A redundant rule performs the same action on the same packets as another rule, such that if the redundant rule is removed, the security policy will not be affected.

III. Dealing with the correction

Once we have detected and identified the existing anomalies, we propose in this section a set of actions in order to correct them and to ensure the coherence of the filter. The detection of firewall anomalies is a well investigated and defined subject. In [18], Al-Shaer and Hamed provided a set of techniques for automatic discovery of firewall policy anomalies. In contrast, when it comes to correcting these anomalies, a complete set of new challenges emerges. In fact, it is likely to introduce new anomalies when trying to correct the existent ones. In order to maintain the coherence of the filter, reordering the rules should take into account the relationships that may exist between them. To achieve the correction task, constraints on the rules order are deduced from the process of anomalies discovery. Based on these constraints, we define a model to arrange the rules with the possibility of adding new rules when needed. To solve the problem we map it to the graph theory domain.

The studies [18] and [2] have defined a set of recommendations for correcting the policy anomalies. However, none of them have defined a clear strategy to put into practice these recommendations. In [2], the author distinguishes between the so-called destructive and constructive anomalies. A similar distinction between anomalies errors and anomalies warnings was made in [18]. In the light of these studies, our scheme will tolerate the existence of the generalization anomaly and we will consider the rest of anomalies as policy errors to resolve. Thus, we allow more relaxed correction actions than the radical disjunctive scheme. In fact, generalization is often used to exclude a specific part of the traffic from a general filtering action. Therefore, we will tolerate the existence of this anomaly. Generalization is considered by [2] as a constructive

anomaly which does not affect the expected filter response unless reordering is made. Shadowing is considered as a critical error in the firewall policy as the shadowed rule never applies, resulting in an unexpected filter response for the packets that match the shadowed rule. Shadowing can be resolved by reversing the relative order of the two involved rules. In this work, we consider redundancy as an error in the firewall policy because a redundant rule adds unnecessary overhead and latency to the filtering process. In the case of redundancy, it is recommended to remove the specific rule which is included in the general rule. Correlation is considered by [18] as an anomaly warning because the correlated rules imply an action that is not explicitly stated by the filtering rules. In the case of correlation, a packet whose header matches the intersection of the headers of the two filtering rules causes ambiguity in packet classification because the adopted policy depends on the order of the two conflicting rules. In this case, the administrator is prompted the order of the rules that complies with his desired policy. In the case of contradiction, the shadowed rule will never be activated. Therefore, we will prompt the administrator to choose which of the two conflicting rules should apply. The other rule will be removed from the filtering rules list since it increases unnecessarily the number of filtering rules.

We propose a set of corrective actions which are inspired from [2]:

Automatic rule's removal: In the case of redundancy, it is recommended to remove the specific rule which is included in the general rule.

Automatic rule's permutation: In the case of shadowing, it is recommended to permute the conflicting rules in order to obtain the anomaly generalization which is tolerated.

Commanded rule's removal: This action is applied in the case of contradiction anomaly. The administrator chooses the rule to be deleted depending on the security policy requirements.

Commanded rule's permutation: In the case of correlation, the administrator chooses the proper order that complies with the adopted policy.

In order to resolve packet conflicts, we define a set of actions divided in three crucial steps. The first step consists of :

1) Resolving the redundancy anomaly.
2) Applying commanded rule's removal in the case of contradiction anomaly.
3) Consulting the administrator in the case of correlation to identify which rule should be stored first in the firewall.

The second step of the anomalies resolution process comprises generating precedence relationships. These relationships are the consequence of generalization, shadowing and correlation. In the third step, a new order of the filtering is performed based on the obtained precedence relationships. Applying this fashion, we deduce a total order of the rules from the local precedence relationships.

A. Precedence relationships

1) Case of generalization: The generalization is considered as a constructive anomaly since it reduces the number of firewall rules. However, this constructive anomaly would turn into shadowing (which is a destructive anomaly) if the relative order of the two conflicting rules was reversed. Hence, the order of the two conflicting rules must be maintained in order to prevent the introduction of the shadowing anomaly. Thus, we add a constraint on the current order in the case of generalisation to prevent shadowing, which is a destructive anomaly. An example of two rules presenting the generalization anomaly is shown graphically in Fig. 1.

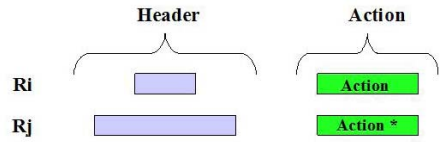

Fig. 1. Generalization anomaly (Action and Action* are two opposite actions).

Thereby, the precedence relationship is $R_i \rightarrow R_j$.

2) Case of correlation : In the case of correlation the administrator chooses which rule has to be stored the first in the filter. This choice is deterministic for the filtering of the packets whose header matches the intersection of the headers of the two correlated rules. In fact, these packets will be treated differently according to which rule will be stored the first in the filter. An illustration of the correlation anomaly is plotted in Fig. 2.

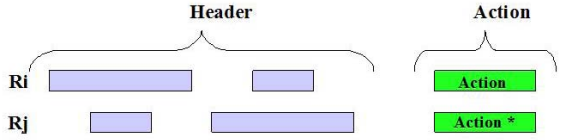

Fig. 2. Correlation anomaly.

Depending on the choice of the administrator the precedence relationship is either $R_i \rightarrow R_j$ or $R_j \rightarrow R_i$. In fact, with regards to Fig. 2, if the administrator chooses $R_i \rightarrow R_j$, then all packets that match the intersection of headers will have the same action as the rule R_i. However, choosing the precedence relationship $R_j \rightarrow R_i$ will lead to the opposite action.

3) Case of shadowing: In the case of shadowing anomaly, the shadowed (or masked) rule will never be activated. To resolve this problem, we invert the order of the two rules in question. Hence, by inverting the two rules, the anomaly shadowing will turn into generalization, which is a tolerable anomaly. Fig. 3 shows two filtering rules presenting the shadowing anomaly.

Hence the precedence relationship is $R_j \rightarrow R_i$.

B. Modeling Precedence relationships

We model each rule by a node of a directed graph. We put a directed edge from the node R_i to the node R_j if R_i precedes

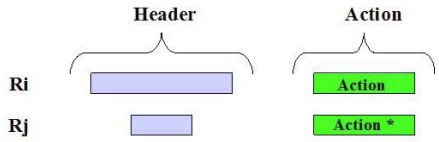

Fig. 3. Shadowing anomaly.

R_j. If the graph associated to the precedence relationships is cyclic, then it is impossible to reorder the rules. Hence, the first step of the correction process is to eliminate the cycles in the graph in order to achieve a topological sort of its nodes.

Theorem 1: If the graph associated to the precedence relationships contains a cycle then there is surely in this cycle a precedence relationship deduced from a correlation anomaly.

Proof:

If the graph associated to the precedence relationships contains a cycle then there exist two rules R_i and R_j verifying R_i precedes R_j and R_j precedes R_i. Let us use reductio ad absurdum: assume that this cycle does not include any precedence relationship deduced from a correlation anomaly. Hence all the relationships in this cycle are deduced from either the generalization or the shadowing anomalies. Therefore, R_i precedes R_j implies that every field of R_i is a subset of the corresponding field of R_j.

On the other hand, R_j precedes R_i implies that every field of R_i is a superset of the corresponding field of R_j. This leads us to conclude that every field of R_i is equal to the corresponding field of R_j. Hence, depending on the actions of the two rules, we will have either a duplicated rule (a particular case of redundancy) or a contradiction anomaly which is impossible because we supposed that we have already eliminated the contradiction and the redundancy anomalies in the first stage of the correction.

■

At this juncture, we shall present a theorem that catalogues some properties of the length of an eventual cycle within the graph modeling the precedence relationships of the firewall.

Theorem 2: The eventual cycles in the directed graph associated to the precedence relationships are even cycles with a length bigger or equal to 4.

Proof:

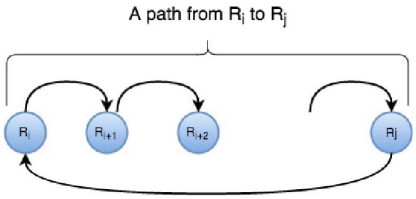

Fig. 4. Graphical representation of the cycle T.

Consider the directed graph $G = (V, E)$, consisting of a set of vertices V and a set of edges E, associated to the precedence relationships. Let R_i and R_j two vertices of V. Suppose that there is a path from R_i to R_j consisted of the sequence of distinct vertices $R_i, R_{i+1}, \ldots, R_j$ so that (R_k, R_{k+1}) for $k = i, \ldots, j-1$ are in E. A path from R_i to R_j together with the edge (R_j, R_i) forms a cycle. Let T be this cycle. A clear representation of T is shown in Fig. 4. To prove the theorem we use reduction absurdum: assume that the cycle T is an odd cycle.

Hence the path from R_i to R_j consists of an even number of edges. Moreover, our precedence relationships are deduced from the anomalies: generalization, correlation and shadowing. Therefore for each edge (R_k, R_{k+1}), $k = i, \ldots, j-1$ the filtering rules R_k and R_{k+1} have distinct actions. If we add the fact that the path from R_i to R_j consists of an even number of edges we can easily prove by recurrence that R_i has the same action as R_j. However, the edge (R_j, R_i), which closes the cycle, implies that R_i and R_j have distinct actions. This is clearly impossible. Consequently, our assumption was wrong: T can not be an odd cycle. Moreover G can not have a cycle of length 2 because two rules can not be exhibit at the same time two kinds of anomalies. So if G has a cycle this cycle will have a length bigger or equal to 4. ■

C. Cycle breaking

In order to perform a topological sort, the graph associated to the precedence relationships should be acyclic. Therefore we have to break the cycles in the graph. In our case, a cycle is due to a precedence relationship deduced from a correlation anomaly. The detection of cycles will be based on the graph algorithm Depth-first search (DFS) [24].

In order to eliminate the cycle, we invert the order of the correlated rules and we add a new rule preceding the two conflicting rules. Let R_x and R_y be the two correlated rules in question where R_x precedes R_y. Thus, the new added rule will have the same action as R_x and every field of this rule is equal to the intersection of the corresponding fields of the two correlated rules.

Fig. 5. Filter 1.

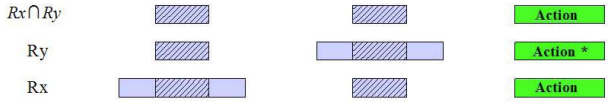

Fig. 6. Filter 2, an equivalent filter to filter 1 (Action and Action* are complementary).

The filter 2 (Fig. 6) is deduced from filter 1 (Fig. 5) by applying the cycle elimination action. It is evident that the two filters are equivalent. In the rest of the paper, we call

intersection filter an added filter to break a cycle. The optimal cycle breaking scheme would result in a minimal number of removed edges and consequently in the smallest number of intersection filter added. However, in our case we can only break the cycles at the edges deduced from the correlation anomaly. Hence, our problem is different from the so called classical Feedback Set Problem [25]. In our implementation, we merely break the cycles at the first traversed edge deduced from a correlation anomaly. Obviously, this may not result in adding a minimum number of intersection filters. Nevertheless, it is possible to combine our algorithm with some heuristics that make the graph acyclic by removing a minimal number of edges.

D. Reordering the rules

The topological sort is an algorithm where the input is a directed acyclic graph and the output is an ordered sequence of vertices. Once we have eliminated the cycles in the graph we apply a topological sort of the nodes. Generally, the topological sort is not unique which makes the reordering flexible.

E. Inserting the intersection filters

Let B be the initial set of filtering rules. We start by applying the topological sort to reorder the filtering rules contained in B. The next step is to insert the eventual added intersection filters into the sorted list of filtering rules. In this section, the proper position of the intersection filter to be inserted should be investigated carefully to avoid creating destructive anomalies. We argue that inserting the intersection filters does not really introduce new anomalies but it reflects existing constructive anomalies. This will be the object of Theorem 3.

Theorem 3: Let R_x and R_y be two correlated rules such as the edge (R_x, R_y) is a part of a cycle. We suppose that this edge is the edge at which the cycle is broken. We denote R_z the resulting intersection filter. Let $A(R_z)$ be the set of rules that present anomalies with R_z and let $A(R_x)$ be the set of rules that present anomalies with R_x apart from redundancy. Then $A(R_z) \subset A(R_x)$.

Proof: Let R_v a rule that presents an anomaly with R_z either than redundancy. Obviously, R_z and R_v have different actions because we supposed that the two rules are not redundant. Consequently, R_v and R_x have different actions. For every field number k, $R_z[k] \bigcap R_v[k] \neq \emptyset$. Moreover, the cycle breaking action implies that for every field k, $R_z[k] \subset R_x[k]$. Therefore, $R_x[k] \bigcap R_v[k] \neq \emptyset$. Since R_x and R_v have different actions, then R_x and R_v are in a conflict different from redundancy because we supposed that we have already eliminated the contradiction and the redundancy anomalies in the first stage of the correction. Therefore $A(R_z) \subset A(R_x)$.

The idea behind theorem 3 is that if the intersection filter R_z is in conflict with an existing filtering rule R_v either than redundancy, then it is because the rule R_x from which the intersection filter R_z is extracted has already an anomaly with R_v. This eventual anomaly between R_x and R_v is necessarily a constructive anomaly because we supposed that we are in the stage of sorting the rule list according to the precedence relationships. In the next sub section, we propose a position of R_z that does not induce a change in the security policy. In this sense, the policy is preserved. In the case where the intersection filter R_z is in redundancy with a rule of B we apply the removal redundancy action.

Position of R_z: In order, to guarantee that introducing R_z does not affect the security policy, we should properly insert R_z in the right position that preserves the policy integrity. From the cycle breaking action, we deduce the following precedence relationship: $R_z \rightarrow R_y \rightarrow R_x$.

Let R_v a rule that presents an anomaly with R_x. Obviously, this anomaly is not redundancy because we supposed that we have eliminated the redundancy anomalies between the rules of B in the first stage of the correction algorithm. The conflict relating R_v to R_x is made constructive by introducing a precedence relationship. The question that arises here, is where to insert R_z in a way that does not jeopardize the constructive anomaly relating R_v and R_x. We will deal with the two cases $Rv \rightarrow R_x$ and $R_x \rightarrow Rv$.

Case 1: $Rv \rightarrow R_x$

We consider first the case where $Rv \rightarrow R_x$. In this case, we propose to insert R_z before R_v by adding this constraint $Rv \rightarrow R_z$ in order to ensure the policy integrity. Before the cycle breaking action, the precedence relationships results in the following order of the rules: R_v, R_x and R_y: $Rv \rightarrow R_x \rightarrow R_y$. By applying the cycle breaking action and considering the constraint $Rv \rightarrow R_z$, we obtain the sorted list of rules $Rv \rightarrow R_z \rightarrow R_y \rightarrow R_x$. In section III-C, we highlighted that the two system of rules $R_x \rightarrow R_y$ and $R_z \rightarrow R_y \rightarrow R_x$ are equivalent and produce the same filtering result. Consequently, the two systems of filtering rules $Rv \rightarrow R_x \rightarrow R_y$ and $Rv \rightarrow R_z \rightarrow R_y \rightarrow R_x$ are equivalent too. Fig. 6 and Fig. 5 illustrate the equivalence of the two systems before cycle breaking and after cycle breaking.

In this perspective, we demonstrate that in the case where $Rv \rightarrow R_x$, it is sufficient to add the constraint $R_z \rightarrow Rv$ in order to guarantee that the added intersection filter does not change the filtering policy.

Case 2: $R_x \rightarrow Rv$

Before the cycle breaking action, the precedence relationships relating the rules R_v, R_x and R_y can be expressed as: $R_x \rightarrow Rv$ and $R_x \rightarrow R_y$. Since R_y and R_v have the same action, then their relative order is not relevant and does not affect the filtering policy. Let us suppose that we arranged the rules in this order: $R_x \rightarrow Rv \rightarrow R_y$ (This arrangement is equivalent to $R_x \rightarrow R_y \rightarrow Rv$).

By applying the cycle breaking action, we obtain the sorted list of rules $R_z \rightarrow R_y \rightarrow R_x \rightarrow Rv$. From section III-C, we know that the two system of rules $R_x \rightarrow R_y$ and $R_z \rightarrow R_y \rightarrow R_x$ are equivalent. Consequently, the two systems of

filtering rules $R_x \rightarrow Rv \rightarrow R_y$ and $R_z \rightarrow R_y \rightarrow R_x \rightarrow Rv$ are equivalent too.

In this case too, we prove that the added intersection filter does not change the filtering policy.

Thus, the proposed insertion mechanism preserves the security policy when adding an intersection filter into the sorted list B of filtering rules. Nevertheless, there is a possibility that an added intersection filter might create redundancy anomaly with an existing rule of B. Therefore the last stage in our algorithm is removing all eventual redundancies that may emerge between the intersection filters and the sorted list of rules B by applying the automatic rule's removal. Such eventual anomaly does not affect the result of the filtering policy but can rather generate an unnecessary processing overhead. ■

F. Remark about the complexity of the correction:

We should underline that the execution time of the anomalies resolution is a minor concern concern. The main performance concern is rather the packet matching-time after deploying the corrected rules. The key contribution of our work is the fact that we do not introduce an excessive increase in the firewall size compared to the related state-of-the-art. It is crucial to keep a compact size of the firewall rules in order to guarantee a low packet matching time. The legacy approaches might render the firewall a bottleneck under high traffic due to an "explosion" in the number of rules as a consequence of the disjunction operation. In addition, the correction is usually performed at rare occasions, namely before to deploying a new firewall policy, thus a reasonable latency can be tolerated before deployment. When it comes to the theoretical complexity of our approach, the cycle detection has a linear complexity according to Tarjan's strongly connected components algorithm [26], namely, the complexity is $O(|V| + |E|)$. The creation of intersection filters is linear in the number of fields of in the header which is simply 4. The topological sort has also a linear complexity as for the case of cycle detection [26]. It is also worth mentioning that Tarjan's strongly connected components algorithm [26] performs also the topological sort as byproduct. Thus, if the rules do not form any cycle, then the topological sort will follow directly without any extra computational complexity. Please note that we ignored the complexity of the anomalies detection process as it is not central in this work. The anomalies detection process can be performed using a myriad of available algorithms from the literature.

IV. Implementation and Computer Experiments

A. The tool

Our work is mainly motivated by the need of a tool that assists network administrators to resolve firewall anomalies. Such a tool was implemented in a user friendly interface which simplifies the analysis and resolution of filtering rules anomalies. We made use of Java as programming language. Comprehensive experiments demonstrated the practicability of our approach and its ability to cope with large sets of filtering rules. In fact, we provided our tool with different firewall configurations and the output was always a set of anomaly free filtering rules. For the sake of brevity we content to present a concise example tested with our tool. The snapshots shown below refer to the example 1 treated in the next subsection: Examples.

Fig. 7 depicts the graphical user interface through it the administrator edits the firewall rules. To start with, the tool alerts the administrator of all eventual anomalies that may exist within the edited filtering rules. The adopted anomaly discovery algorithm is outlined in [18].

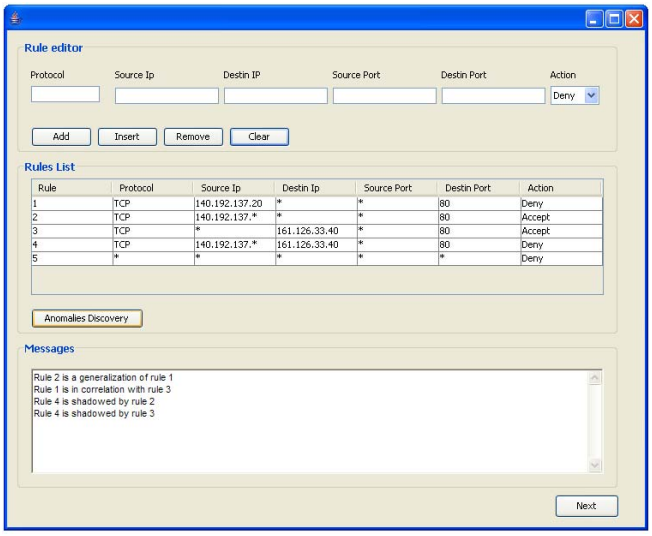

Fig. 7. Reporting the detected anomalies.

The tool analyses the filtering rules and reports the detected anomalies to the user. In Fig. 8, the administrator is guided through message prompts to choose the proper actions that comply with his desired policy.

A proper order of the filtering rules is then performed by applying our paradigm. The filter after correction is shown graphically in Fig. 9.

B. Examples

In order to explain our paradigm in an easy way we use two examples.

1) Example 1: We consider the below list of filtering rules including some anomalies. Note that this example was treated above by our tool as depicted in Fig. 7.

Rule number	Protocol	Source Address	Destination Address	Source Port	Destination Port	Action
1	TCP	140.192.137.20	*	*	80	Deny
2	TCP	140.192.137.*	*	*	80	Accept
3	TCP	*	161.126.33.40	*	80	Accept
4	TCP	140.192.137.*	161.126.33.40	*	80	Deny
5	*	*	*	*	*	Deny

The anomalies contained within this set of filtering rules are:

1) Rule 2 is a generalization of rule 1.
2) Rule 1 is in correlation with rule 3.

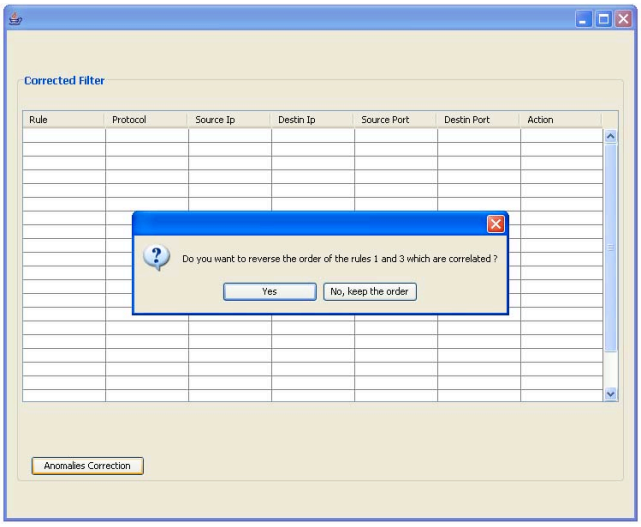

Fig. 8. Message prompting the user to choose the proper corrective action.

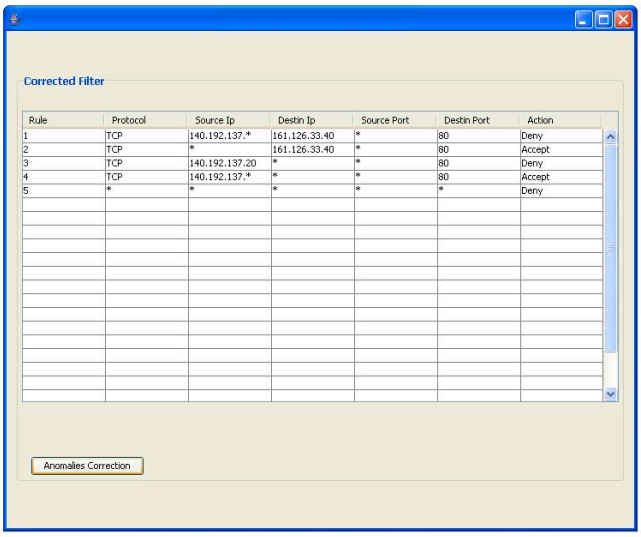

Fig. 9. Proper Order.

3) Rule 4 is shadowed by rule 2.
4) Rule 4 is shadowed by rule 3.

In the case of correlation, we suppose that the administrator chooses to apply the rule 3 before 1. Subsequently, the precedence relationships can be summarized in the table below:

R_x	R_y	Anomaly	Precedence relationship
1	2	Generalization	$1 \rightarrow 2$
1	3	Correlation	$3 \rightarrow 1$
4	2	Shadowing	$4 \rightarrow 2$
4	3	Shadowing	$4 \rightarrow 3$

In order to arrange the rules we perform Depth-first search. The right order that guarantees the coherence of the filter is described in Fig. 10. Fig. 9 shows that the same result is

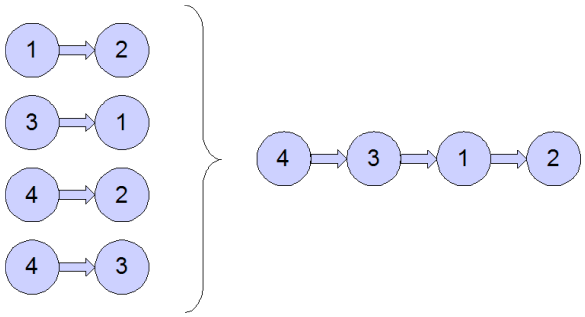

Fig. 10. The graph associated to the precedence relationships in Example 1.

achieved by the tool. Subsequently, the filter after correction is given in the table below:

Rule number	Protocol	Source Address	Destination Address	Source Port	Destination Port	Action
4	TCP	140.192.137.*	161.126.33.40	*	80	Deny
3	TCP	*	161.126.33.40	*	80	Accept
1	TCP	140.192.137.20	*	*	80	Deny
2	TCP	140.192.137.*	*	*	80	Accept
5	*	*	*	*	*	Deny

2) Example 2: We consider the below list of filtering rules including some anomalies.

Rule number	Protocol	Source Address	Destination Address	Source Port	Destination Port	Action
1	TCP	140.192.*.*	*	*	*	Deny
2	TCP	140.192.24.*	*	Port<1024	80	Accept
3	TCP,UDP	140.192.37.*	161.126.33.*	Port<1024	80,53	Accept
4	TCP,UDP	140.192.*.*	161.126.*.*	*	*	Deny
5	*	*	*	*	*	Deny

First, we identify the anomalies within the filtering rules. This process detects the following anomalies:

1) Rule 1 shadows rule 2.
2) Rule 1 is in correlation with rule 3.
3) Rule 4 is in correlation with rule 2.
4) Rule 4 is a generalization of rule 3.

We suppose that we prompted the administrator to choose the proper order in the case of correlation and we assume that he chose to apply 1 then 3 and to apply 4 then 2. The precedence relationships are then described in the table below:

R_x	R_y	Anomaly	Precedence relationship
1	2	Shadowing	$2 \rightarrow 1$
1	3	Correlation	$1 \rightarrow 3$
4	2	Correlation	$4 \rightarrow 2$
3	4	Generalization	$3 \rightarrow 4$

The graph associated to the precedence relationships is shown in Figure 11.

In order to perform a topological sort we have to make the directed graph acyclic by breaking the cycle $2 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 2$. We choose to break the cycle at the edge $(4, 2)$ which is the consequence of a correlation anomaly. Therefore we should add the rule to keep the coherence of the filter. The next two tables below illustrate the case of the rules 4 and 2 before correction and after correction (order inversion and introduction of intersection filter).

Subsequently, the filter after correction is given in the Table I.

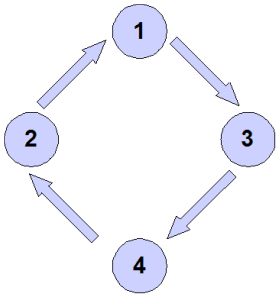

Fig. 11. The graph associated to the precedence relationships in Example 2.

4	TCP,UDP	140.192.*.*	161.126.*.*	*	*	Deny
2	TCP	140.192.24.*	*	Port<1024	80	Accept

	TCP	140.192.24.*	161.126.*.*	Port<1024	80	Deny
2	TCP	140.192.24.*	*	Port<1024	80	Accept
4	TCP, UDP	140.192.*.*	161.126.*.*	*	*	Deny

Rule number	Protocol	Source Address	Destination Address	Source Port	Destination Port	Action
	TCP	140.192.24.*	161.126.*.*	Port<1024	80	Deny
2	TCP	140.192.24.*	*	Port<1024	80	Accept
1	TCP	140.192.*.*	*	*	*	Deny
3	TCP,UDP	140.192.37.*	161.126.33.*	Port<1024	80,53	Accept
4	TCP,UDP	140.192.*.*	161.126.*.*	*	*	Deny
5	*	*	*	*	*	Deny

TABLE I
RESULT OF THE CORRECTION PROCESS FOR EXAMPLE 2.

V. CONCLUSION

Firewall rules misconfiguration is a substantial issue in the area of network security. Valuable studies in this field have provided an answer to the anomalies discovery issue. However, the correction of these anomalies is still a rich axis of research. This task is extremely delicate and poses serious challenges with regards to the importance of rules order. In this paper, we have proposed a new approach for correcting anomalies within filtering rules. The correction is assisted by the administrator to yield a required precision in reflecting the adopted security policy. The heart of our scheme involves introducing precedence relationships in order to perform a global order of the filtering rules. We have implemented our method and the results are very promising. The implementation provides the network administrators with a valuable tool that simplifies greatly the anomalies resolution process and avoids the manual correction which is a tedious and error prone task. As a future work, we are considering extending of our approach in order to handle anomalies in distributed firewalls [27].

REFERENCES

[1] A. Hari, S. Suri, and G. Parulkar, "Detecting and resolving packet filter conflicts," in *Proceedings of INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3. IEEE, 2000, pp. 1203–1212.

[2] A. Bouhoula, Z. Trabelsi, E. Barka, and M.-A. Benelbahri, "Firewall filtering rules analysis for anomalies detection," *International Journal of Security and Networks*, vol. 3, no. 3, pp. 161–172, 2008.

[3] Y. Bartal, A. Mayer, K. Nissim, and A. Wool, "Firmato: A novel firewall management toolkit," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy*. IEEE, 1999, pp. 17–31.

[4] J. D. Guttman, "Filtering postures: Local enforcement for global policies," in *Proceedings of the 1997 IEEE Symposium on Security and Privacy*. IEEE, 1997, pp. 120–129.

[5] S. Hinrichs, "Policy-based management: Bridging the gap," in *Proceedings of the 15th Annual Conference on Computer Security Applications (ACSAC'99)*. IEEE, 1999, pp. 209–218.

[6] A. Mayer, A. Wool, and E. Ziskind, "Fang: A firewall analysis engine," in *Proceedings of the 2000 IEEE Symposium on Security and Privacy, 2000. S&P*. IEEE, 2000, pp. 177–187.

[7] A. Wool, "Architecting the lumeta firewall analyzer." in *USENIX Security Symposium*, 2001, pp. 85–97.

[8] P. Eronen and J. Zitting, "An expert system for analyzing firewall rules," in *Proceedings of the 6th Nordic Workshop on Secure IT Systems (NordSec 2001)*, 2001, pp. 100–107.

[9] C. Pornavalai and T. Chomsiri, "Firewall policy analyzing by relational algebra," in *The 2004 International Technical Conference on Circuits/Systems, Computers and Communications*, 2004.

[10] L. Yuan, H. Chen, J. Mai, C.-N. Chuah, Z. Su, and P. Mohapatra, "Fireman: A toolkit for firewall modeling and analysis," in *2006 IEEE Symposium on Security and Privacy*. IEEE, 2006, pp. 15–pp.

[11] A. Saadaoui, N. B. Y. B. Souayeh, and A. Bouhoula, "Automated and optimized fdd-based method to fix firewall misconfigurations," in *14th IEEE International Symposium on Network Computing and Applications, NCA 2015, Cambridge, MA, USA, September 28-30, 2015*, 2015, pp. 63–67.

[12] T. Abbes, A. Bouhoula, and M. Rusinowitch, "An inference system for detecting firewall filtering rules anomalies," in *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 2008, pp. 2122–2128.

[13] ——, "Detection of firewall configuration errors with updatable tree," *International Journal of Information Security*, pp. 1–17, 2015.

[14] C. Benecke, "A parallel packet screen for high speed networks," in *Proceedings of the 15th Annual Conference on Computer Security Applications (ACSAC'99)*. IEEE, 1999, pp. 67–74.

[15] H. Hamed, A. El-Atawy, and E. Al-Shaer, "Adaptive statistical optimization techniques for firewall packet filtering," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, April 2006, pp. 1–12.

[16] ——, "On dynamic optimization of packet matching in high-speed firewalls," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1817–1830, 2006.

[17] D. Eppstein and S. Muthukrishnan, "Internet packet filter management and rectangle geometry," in *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2001, pp. 827–835.

[18] E. S. Al-Shaer and H. H. Hamed, "Firewall policy advisor for anomaly discovery and rule editing," in *IFIP/IEEE Eighth International Symposium on Integrated Network Management*. IEEE, 2003, pp. 17–30.

[19] A. K. Bandara, A. Kakas, E. C. Lupu, and A. Russo, "Using argumentation logic for firewall policy specification and analysis," in *Large Scale Management of Distributed Systems*. Springer, 2006, pp. 185–196.

[20] F. Cuppens, N. Cuppens-Boulahia, and J. G. Alfaro, "Detection of network security component misconfiguration by rewriting and correlation," in *Joint Conference on Security in Network Architectures and Security of Information Systems*, 2006.

[21] J. García-Alfaro, F. Cuppens, and N. Cuppens-Boulahia, "Towards filtering and alerting rule rewriting on single-component policies," in *Computer Safety, Reliability, and Security*. Springer, 2006, pp. 182–194.

[22] N. B. Y. B. Souayeh and A. Bouhoula, "A fully automatic approach for fixing firewall misconfigurations," in *2011 IEEE 11th International Conference on Computer and Information Technology (CIT)*. IEEE, 2011, pp. 461–466.

[23] E. Al-Shaer and H. Hamed, "Design and implementation of firewall policy advisor tools," *DePaul University, CTI, Tech. Rep*, 2002.

[24] A. Bundy and L. Wallen, "Depth-first search," in *Catalogue of Artificial Intelligence Tools*. Springer, 1984, pp. 29–29.

[25] P. Festa, P. M. Pardalos, and M. G. Resende, "Feedback set problems," in *Handbook of combinatorial optimization*. Springer, 1999, pp. 209–258.

[26] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM journal on computing*, vol. 1, no. 2, pp. 146–160, 1972.

[27] E. S. Al-Shaer and H. H. Hamed, "Discovery of policy anomalies in distributed firewalls," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4. IEEE, 2004, pp. 2605–2616.