An optimized firewall anomaly resolution

# An Optimized Firewall Anomaly Resolution

Fulvio Valenza[1]* and Manuel Cheminod[2]
[1]Politecnico di Torino, DAUIN, corso duca degli Abruzzi 24, Turin, Italy
fulvio.valenza@polito.it
[2]CNR-IEIIT, corso duca degli Abruzzi 24, Turin, Italy
manuel.cheminod@ieiit.cnr.it

**Abstract**

Firewalls are the key mechanism in cybersecurity, that has been widely used to ensure network security. In literature, several works have been proposed in the area of firewall rules managing, however, the correct firewall configuration still remains a complex and error-prone task. Anomalies among firewall rules can cause severe network breaches, such as allowing harmful packets to slip into a subnetwork or dropping legitimate traffic which in turn could hinder the correct availability of web services. This paper aims to help the network security administrators by introducing a formal approach that reduces the number of anomalies in firewalls' configurations that the administrators are usually obligated to manually solve.

**Keywords**: Firewall, Policy Based Systems, Policy Anomaly Management, Network Security

## 1 Introduction

Firewalls are known as a main architectural element for the security of every IT system. The last Verizon report clearly shows [25] that about 70% of cyber attacks discovered in 2019 had network/cloud services and resources as main targets. Firewalls have been widely used as the very first frontier to protect not only small individual and local networks, but also large networks from these cyberattacks.

Moreover, nowadays firewalls are not used as perimetral defences only but are more and more adopted to protect internal layers in large networks, for instance in industrial networks and critical infrastructures, where defence in depth is required.

Unfortunately, the configuration of firewalls is mostly designed manually by network administrators, and the support of automatic or semi-automatic tools for this task, is limited. Of course, in this scenario, the possibility of introducing human errors in such configurations is high and this can have a great impact on the effectiveness of the firewall in providing an adequate security and protection level. This is even more critical in large networks and virtualized environments. Specifically, in large networks many security mechanisms are in place and flaws in a firewall configuration could easily propagate through the entire network. While in virtualized networks, the paths that traffic flows must cross can be defined at run-time by means of software programs (i.e., Software Defined Network) [10], and the network functions can be virtualized so as to be deployed at on-demand on general-purpose servers (i.e., Network Functions Virtualization) [16].

In literature, several works have been proposed to automatically detect anomalies in a firewall configuration [1, 11, 12, 19–24, 26, 28]. Moreover, some of the proposed works are also able to partially fix the detected anomalies, notwithstanding, automatic conflict resolutions are potentially very dangerous

*Journal of Internet Services and Information Security (JISIS)*, volume: 10, number: 1 (February 2020), pp. 22-37
DOI: *10.22667/JISIS.2020.02.29.022*

*Corresponding author: Fulvio Valenza, Politecnico di Torino, Dip. di Automatica e Informatica, Corso Duca degli Abruzzi, 24, 10129 Torino , Tel: +39-(0)11-090-7026

| | priority | IPsrc | Psrc | IPdst | Pdst | Proto | Action |
|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 130.162.0.1 | * | * | 80 | TCP | ALLOW |
| $r_2$ | 2 | 130.162.0.0/24 | * | | 80 | TCP | DENY |
| $r_3$ | 3 | 130.162.0.1/24 | * | 130.162.0.2/24 | * | TCP | ALLOW |
| $r_4$ | 4 | 130.162.0.1/24 | * | 130.162.0.2/24 | * | UDP | ALLOW |
| $r_5$ | 5 | * | * | 130.162.3.1 | * | * | DENY |
| $r_6$ | 6 | * | * | 130.162.3.0/24 | 0-1024 | TCP | ALLOW |
| $r_7$ | 7 | * | * | 130.162.3.0/24 | * | * | DENY |
| ... | | | | | | | |
| default | $\infty$ | * | * | * | * | * | DENY |

Figure 1: Example of packet filter rule set (without anomalies)

because they actually hide mistakes and conflicting requirements that, on the contrary, should be explicitly addressed by security administrators. For these reasons network security administrators prefer to manually solve firewall anomalies. However, the number of anomalies in firewalls are huge, since a firewall policy may consist of hundreds of rules, which are typically logically correlated with each other.

This paper aims to define a formal approach to select a minimum set of anomalies that, if solved, will result in the complete resolution of all the initially identified anomalies among the firewall rules.

The reduction of anomalies to solve, significantly simplifies and makes more effective the security administrator work, improving the overall quality and efficiency of the firewall configuration and thus increasing the global security of the network. Moreover, a correct firewall configuration improves network performance, especially in the time-driven networks [2, 3].

Specifically, we propose a model suitable to identify the relations among the fields of the firewall rules and among the rules themselves (*intra-rule* and *inter-rules*). These relations are then used to point out the anomalies and to reduce them to a minimum set. Moreover, in this work we propose a novel classification of firewall rule anomalies that extends and improves the previous works.

The remainder of the paper is structured as follows. Section 2 gives an overview of firewall operations, rules and anomalies, and the current state of the arts in this area. Section 3 presents the policy model, where we describe the proposed rule relations and the novel firewall anomaly taxonomy. Section 4 and 5 describes the optimized anomaly resolution approach that we followed in this paper, and its implementation and validation. Finally, Section 6 provides a brief conclusion of the paper.

# 2 Background

Before the description of the proposed approach, in this section we briefly introduce the background on firewall operation and the policy-base managing.

## 2.1 Firewall

Firewalls are network security controls that regulate the traversal of packets by the definition of: (i) an ordered set of rules; (ii) a default action; (iii) a resolution strategy [14].

According to RFC-3198 [27], a *Rule r* is composed by a set of conditions $C$ and one action $a$.

$$r = (C, a)$$

Each firewall condition, belonging to the $C$ set, can be a single or a range of values and represents the possible values of the corresponding field in actual packets which match this rule. The Firewall actions

| | priority | IPsrc | Psrc | IPdst | Pdst | Proto | Action |
|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 130.162.0.1 | * | 130.162.0.2 | 80 | TCP | ALLOW |
| $r_2$ | 2 | 130.162.1.3 | * | 130.162.2.2 | * | UDP | ALLOW |
| $r_3$ | 3 | 130.162.0.1 | * | 130.162.0.2 | 80 | TCP | DENY |
| $r_4$ | 4 | 130.162.1.0/24 | * | 130.162.2.0/24 | * | * | ALLOW |
| $r_5$ | 5 | 130.162.1.0/24 | * | 130.162.2.0/24 | 80 | * | ALLOW |
| $r_6$ | 6 | 130.162.3.0/24 | * | 130.162.0.0/24 | * | * | DENY |
| $r_7$ | 7 | 130.162.3.1 | * | 130.162.0.1 | 80 | TCP | ALLOW |
| $r_8$ | 8 | 130.162.1.1 | * | 130.162.2.1 | 22 | TCP | ALLOW |
| $r_9$ | 9 | 130.162.1.4 | * | 130.162.2.5 | 0-1024 | * | DENY |
| $r_{10}$ | 10 | 130.162.3.0/24 | * | 130.162.0.0/24 | * | * | DENY |
| $r_{11}$ | 11 | 130.163.3.0/24 | * | 130.162.0.0/24 | * | * | DENY |
| ... | | | | | | | |
| default | $\infty$ | * | * | * | * | * | DENY |

Figure 2: Example of packet filter rule set with anomalies

can be *allow*, which forwards the packet, or *deny*, which discards the packet. Thus, the packet is allowed or denied by a specific rule if the packet header matches all the conditions of this rule. The *default action* is the action (i.e., allow or deny) to apply if the packet does not meet all the conditions of any rule.

Finally, the *resolution strategy* describes which rule's action should be applied if more than one rule matches the packet. Some example of resolution strategies are:

- *First Matching Rule* (FMR) selects the action from the first applicable rule in an ordered list;
- *Allow Takes Precedence* (ATP) where in case of contradicting actions that are simultaneously activated we enforce the Allow rule over the Deny one;
- *Deny Takes Precedence* (DTP) where in case of contradicting actions that are simultaneously activated we enforce the Deny rule over the Allow one (this is a restrictive strategy);
- *Most Specific Takes Precedence* (MSTP) where in case two conflicting rules are applied, the most specific rule is the one that takes precedence;
- *Least Specific Takes Precedence* (LSTP) where in case two conflicting rules are applied, the less specific rule is the one that takes precedence.

Without loss of generality, in this work, we consider the packet filter, the most common and used type of firewall, with deny as default action and FMR as resolution strategy. Specifically, in the packet filter condition fields are: source IP address, source port, destination IP address, destination port and protocol type (as show in the example in FIGURE 1).

$$r = (IPsrc, Psrc, IPdst, Pdst, Proto, Action)$$

## 2.2 Policy-based management

The application of policy-based management in network security has received increasing attention by the scientific community for many years, as reported in [8, 15]. The research on this topic mainly focuses on: *policy analysis*, *policy refinement* and *policy verification*.

Policy analysis checks the inconsistencies or sub-optimization in the specification of policy rule sets [23]. Policy analysis only concerns the policy specification and does not check if and how policies are enforced in the networks.

Policy refinement bridges the gap between specification and implementation. A policy refinement process, indeed, translates high-level requirements into low-level configurations (i.e., the policy rules) [6].

Finally, policy verification checks if a set of security properties/requirements are correctly enforced by low-level configurations of the security controls in the system. It is typically used to validate a hand-made security configuration [9].

A common example of security property is network reachability, i.e., verify which kind of packets can travel from one node to another [5].

In this work we focus on the policy analysis. Several works can be found in the literature concerning policy analysis and the main contributions in this area mostly deal with *anomaly analysis* of firewall and VPN policies.

Anomaly analysis[1] looks for incorrect policy specifications that a network/security administrator may introduce. Anomaly analysis detects potential conflicts, errors and sub-optimizations affecting single or multiple policy sets [23].

One of the most influential works in this area is by Al-Shaer et al., which addresses the analysis of filtering configurations via First-Order Logic (FOL) formulas [1]. The authors classified and analysed both the local anomalies arising in a single firewall (intra-firewall anomaly) and the global ones taking into account several distributed filtering devices (inter-firewall anomaly).

Liu et al. focused on detecting and removing redundant filtering rules with data-structure named FDD (Firewall Decision Diagram) [21]. The authors distinguish upward redundant rules, which are rules that are never matched, and downward redundant rules, which are rules that are matched but enforce the same action as some lower priority rules.

Valenza et al. define and detect anomalies among different security functions (inter-function anomalies) [4, 24]. The authors proposed a formal model able to detect several kinds of errors and anomalies that originate from correlations between configuration rules of different network functions. Their approach is able to cope with a wide array of different network functions, such as firewalls, NAT/NAPT devices, traffic monitors and encryption devices. In particular, they have presented three macro types of anomalies: blocked traffic, modified traffic and encrypted traffic.

Hu et al. [13], [12], propose FAME (Firewall Anomaly Management Environment) a visualization-based firewall policy manager. This tool identifies policy anomalies and derives a possible resolution, by a rule-based segmentation technique and a grid-based representation. In our view, the automatic resolution the identify policy anomalies can potentially alter the original behaviours of the security administrators.

Basumatary et al. [7] propose a formal model for firewall anomaly detection. They represent firewall rules by a topological-temporal model and use a model checker to verify the firewall policy.

Krombi et al. [17, 18] propose a procedure that synthesizes an automaton that implements a given security policy. They use our automaton to verify completeness, detect anomalies, and discover functional discrepancies between several implementations of a security policy.

## 3 Policy Model

In this section, we formally present the firewall rule relations enhancing the work described in [24], then we report the firewall policy anomalies extending the taxonomy presented in [1].

[1] In some works, anomaly analysis is called as either policy validation or conflict analysis.

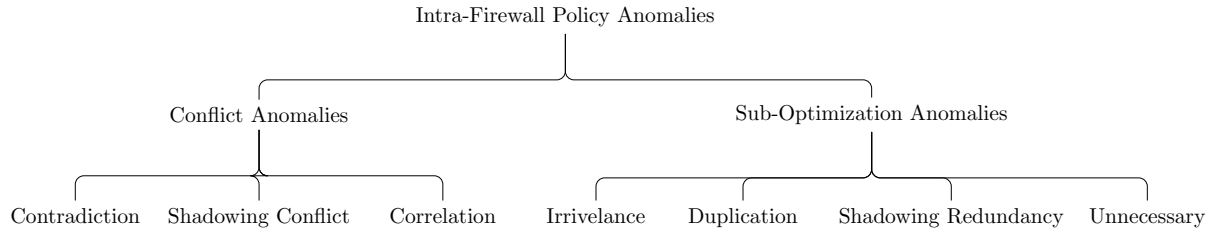



Figure 3: Taxonomy of Intra-Firewall Policy Anomalies

### 3.1 Rule Relations

A correct model of rule relations is necessary for the analysis and detection of anomalies, derived from an erroneous specification. Our model, specifically, supports four types of relations between the condition fields (i.e., source IP address, destination IP address, source port, and destination port and protocol type), that are:

- **equivalence** ($f_x = f_y$): two condition fields $f_x$ and $f_y$ are equivalent if they have the same value (or range of values);

- **dominance** ($f_x \succ f_y$): a condition field $f_x$ dominates another one $f_y$ if it is a generalization of the second one. For example, $f_x$ is the source IP addresses 10.10.*.* and $f_y$ is the source IP address 10.10.10.*, in this case $f_x$ dominates $f_y$. The symbol $*$, called wildcards, allows to define a set or interval of values (e.g., 10.10.11.$*$ stands for 10.10.11/24, whereas if $*$ refers to a port field it represents all the possible ports from 0 to 65535).

- **correlation** ($f_x \sim f_y$): two condition fields $f_x$ and $f_y$ are correlated if they share some values, but none of them includes (or dominates) the other one. For example, if $f_x$ and $f_y$ are destination port and $f_x$ value is 25-65, while $f_y$ value is 40-75, then they are correlated because the range 40-65 is shared by both fields;

- **disjointness** ($f_x \perp f_y$): two condition fields $f_x$ and $f_y$ are disjoint if they do not share any value. On the other hand, if a network field is equivalent, correlated or dominates another, those fields are *not-disjointed* ($f_x \not\perp f_y$).

Having defined the relations among fields we proceed with the definition of relations among two condition sets, $C_x$ and $C_y$. In particular, we define four possible relations among two conditions: *equivalence*, *dominance*, *correlation*, *disjointness*.

Please note that here we define the meaning of relations between conditions (e.g., $C_x$ and $C_y$) using relations between corresponding condition fields ($f_x^i$ and $f_y^i$).

Given two conditions $C_x$ and $C_y$, only one of the following relations holds:

- **equivalence**: two conditions $C_x$ and $C_y$ are equivalent (or equal) if each condition field $f_x^i$ in $C_x$ is equivalent to the corresponding condition element $f_y^i$ in $C_y$ so that they exactly match the same packets (headers);

$$C_x \equiv C_y \Leftrightarrow f_x^i = f_y^i \forall i$$

  For example, referring on the rules set in FIGURE 2, there is equivalence between the conditions of the rule $r_1$ and $r_3$.

- **dominance**: a condition $C_x$ dominates $C_y$ ($C_x \succ C_y$) if it is a generalization of the latter. In other words, if the first condition set matches all the packet matched by the second, *and some more* (otherwise this would be an equivalence relation).

$$C_x \succ C_y \Leftrightarrow C_x \not\equiv C_y \wedge f_x^i \succeq f_y^i \forall i$$

  where $f_x \succeq f_y$ stands for $f_x \succ f_y \vee f_x \equiv f_y$.

  For example, the condition of rule $r_4$ dominates the conditions of the rule $r_5$.

- **correlation**: two conditions $C_x$ and $C_y$ are correlated ($C_x \sim C_y$) if they match some common packets, but none of them includes (or dominates) the other one.

$$C_x \sim C_y \Leftrightarrow C_x \not\succeq C_y \wedge C_x \not\preceq C_y \wedge \forall i | f_x^i \not\perp f_y^i$$

  where $C_x \not\succeq C_y$ stands for $C_x \not\succ C_y \wedge C_x \neq C_y$. For example, the conditions of rule $r_5$ and $r_9$ are correlated.

- **disjointness**: two conditions are disjoint if they do not match any common packet.

$$C_x \perp C_y \Leftrightarrow \exists i | f_x^i \perp f_y^i$$

  Please note that $C_x \not\perp C_y$ means that $C_x$ and $C_y$ are either equivalent, correlated or one dominates the other (i.e, $\equiv, \succ, \sim$).

  For example, the conditions of rule $r_1$ and $r_9$ are disjoint.

Finally, the priority of a rule $r$ is here represented with the function $\pi(r)$. Specifically, the function $\pi(r)$, returns the position of $r$ in the ordered rule set. In this work we put at the top of the list the rules with the highest priority. In this way, between any two rules $r_x$ and $r_y$ it exists either the relation $\pi(r_x) > \pi(r_y)$ or the opposite $\pi(r_x) < \pi(r_y)$.

## 3.2 Anomalies

Policy anomalies typically occur when multiple authors defining the set of policy rules. Moreover, the modification or creation of a policy rule is a difficult task, because a new/updated policy can affect the behaviour of existing policies, defined by other people at different times. When firewalls contain a large number of rules, the risk of writing anomalies is significantly high [15].

In literature, there are two types of firewall policy anomaly: *intra-policy* and *inter-policy*[2].

An *intra-policy* anomaly is an anomaly among two rules in the same policy set (i.e., two rules of the same firewall), while an *inter-policy* anomaly is an anomaly among two rules in two different policy set (i.e., two rules of different firewalls).

In this work, we focus only on *intra-policy* anomalies where an optimized and effective firewall anomaly resolution is needed, as better described in Section 4.

As shown in FIGURE 3, we distinguish two other main types of firewall policy anomaly, which are:*conflict* anomalies and *sub-optimization* anomalies.

A *sub-optimization* anomaly arises when redundant rules or other less efficient policy implementations are present. A *conflict* anomaly arises when the effect of one rule is influenced or altered by another one, e.g. the actions of two rules (that are both satisfied simultaneously) contradict each other. Typically a conflict occurs when a set of policies rules (two or more) are simultaneously satisfied. This implies that

[2]Note that in some works, the *intra-policy* anomalies are also called as *intra-firewall*, while the *inter-policy* as *inter-firewall*.

the combined actions of the rules may produce different results depending on the order of execution of these actions. This is because the filtering procedure is performed by sequentially matching the packet against firewall rules until a match is found. Obviously, if the firewall rules are disjoint, the order of the rules does not influence the overall final behaviour.

Formally, given two rules $r_x = (C_x, a_x)$, $r_y = (C_y, a_y)$ with some relation (i.e., $C_x \not\perp C_y$), we have a conflict when $a_x \neq a_y$, otherwise we have a sub-optimization.

In our view, it is always possible to automatically solve the *sub-optimization* anomalies by applying some specific resolution strategy. *Conflict* anomalies, on the contrary, have to be manually solved by network administrators since it is not possible to define a single automatic resolution strategy that is valid in all the possible cases and that can solve all these anomalies, without potentially changing the behaviour of the related security policies.

In the following, we formally describe each intra-firewall anomaly and the possible action that a security/network administrator can perform in order to solve each anomaly.

### 3.2.1 Sub-optimization anomalies

As shown in FIGURE 3, we divide the *sub-optimizations* anomalies in four types: *Irrelevance*, *Duplication*, *Shadowing Redundancy*, *Unnecessary*.

**Irrelevance**

A policy rule $r_x$ is irrelevant (i.e., $\mathscr{A}_{irr}(r_x)$) if it does not match any packet that might arrive to the firewall. This occurs when either the source or the destination address of the rule does not match with the subnet protected by the firewall. Given the set $R$ of firewall rules, we represent the set of all the irrelevant rules as $\mathbb{A}_{irr}$:

$$\mathbb{A}_{irr} = \{r \in R | \mathscr{A}_{irr}(r)\}$$

Cancellation of an irrelevant policy rule does not alter the policy behaviour, while it will increase the system performance (as, in general, a large number of rules that are evaluated but never applied can hinder the performance of the filtering system).

For example, we can consider as irrelevant the rule $r_{11}$ in the FIGURE 2, because subnet $130.163.0.0/16$ is not under the protection of the firewall where $r_{11}$ is deployed.

**Duplication Anomaly**

A policy rule $r_x$ duplicates rule $r_y$ and vice versa if they specify the same action and match the same packets:

$$\mathscr{A}_{dupl}(r_x, r_y) \coloneqq C_x \equiv C_y \wedge a_x = a_y \qquad \mathbb{A}_{dupl} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{dupl}(r_x, r_y)\}$$

In this case, the removal of the rule with the lowest priority $\pi$ between $r_x$ and $r_y$ will not change the policy behaviour.

Referring to the example in FIGURE 2, there is a duplication anomaly between rules $r_6$ and $r_{10}$.

**Shadowing Redundancy Anomaly**

A policy rule $r_y$ is shadowed by rule $r_x$ if $\pi(r_x) > \pi(r_y)$ and all packets matched by $r_y$ are also matched by $r_x$. In this case $r_y$ will never be applied.

We specify the anomaly as a *Shadowing Redundancy* anomaly when $r_x$ and $r_y$ specify the same action:

$$\mathscr{A}_{shaRed}(r_x, r_y) \coloneqq \pi(r_x) > \pi(r_y) \wedge C_x \succ C_y \wedge a_x = a_y \qquad \mathbb{A}_{shaRed} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{shaRed}(r_x, r_y)\}$$

In this case, the removal of $r_y$ (i.e., the rule with the lower priority) is possible without changing the overall policy behaviour.

In the above example, we can find a Shadowing Redundancy anomaly between rules $r_4$ and $r_5$, $r_4$ and $r_8$.

**Unnecessary Anomaly**

A policy rule $r_x$ is redundant (i.e., unnecessary) with respect to rule $r_y$ when: (i) $r_x$ and $r_y$ specify the same action, (ii) $\pi(r_y) < \pi(r_x)$ ($r_x$ precedes $r_y$), (iii) all packets that are matched by $r_x$ also matched by $r_y$, and (iv) it does not exist a rule $r_z \not\perp r_x$ with a priority between $r_x$ and $r_y$ ($\pi(r_y) < \pi(r_z) < \pi(r_x)$) and with opposite action.

$$\mathscr{A}_{unn}(r_x, r_y) := \pi(r_x) > \pi(r_y) \wedge C_y \succ C_x \wedge a_x = a_y \wedge \nexists r_z | \pi(r_y) < \pi(r_z) < \pi(r_x) \wedge C_z \not\perp C_x \wedge a_z \neq a_x$$
$$\mathbb{A}_{unn} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{unn}(r_x, r_y)\}$$

In this case, the removal of $r_x$ (i.e., the rule with higher priority) will not change the policy behaviour.
You can see an unnecessary anomaly between rule $r_2$ and $r_4$.

### 3.2.2 Conflict Anomalies

We further classify *conflict* anomalies in three types: *Contradiction*, *Shadowing Conflict*, *Correlation*.

**Contradiction Anomaly**

Two policy rules $r_x$ and $r_y$ are in contradiction if they match the same packets but they specify different actions.

$$\mathscr{A}_{con}(r_x, r_y) \ := \ C_x \equiv C_y \ \wedge \ a_x \neq a_y \qquad \mathbb{A}_{con} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{con}(r_x, r_y)\}$$

An automatic removal of this anomaly is not possible, and an evaluation and action from the administrator is needed. Specifically, the administrator must decide which rule should be removed.

There is contradiction between the rule $r_1$ and $r_3$.

**Shadowing Conflict Anomaly**

As mentioned before, a policy rule $r_y$ is shadowed by rule $r_x$ if $\pi(r_x) > \pi(r_y)$ and all packets matched by $r_y$ are also matched by $r_x$. In this case $r_y$ will never be applied.

We specify the anomaly as a *Shadowing Conflict* anomaly when the two rules specify different actions.

$$\mathscr{A}_{shaConf}(r_x, r_y) \ := \ \pi(r_x) > \pi(r_y) \ \wedge \ C_x > C_y \ \wedge \ a_x \neq a_y \qquad \mathbb{A}_{shaConf} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{shaConf}(r_x, r_y)\}$$

Also, in this case, the automatic resolution of this anomaly is not possible and the administrator must evaluate case by case. Specifically, in order to solve the anomaly, the administrator can decide to: (i) remove $r_y$; or (ii) put $r_y$ just before $r_x$[3].

A shadowing conflict anomaly is found between rule $r_6$ and $r_7$.

**Correlation Anomaly**

Two policy rules are correlated when they specify different actions and some packets that are matched by $r_x$ are also matched by $r_y$ but there are other packets that are either matched by $r_x$ only or by $r_y$ only.

$$\mathscr{A}_{corr}(r_x, r_y) \ := \ C_x \sim C_y \ \wedge \ a_x \neq a_y \qquad \mathbb{A}_{corr} = \{(r_x, r_y) \in R^2 | \mathscr{A}_{corr}(r_x, r_y)\}$$

In order to solve a correlation anomaly, the administrator can decide to: (i) leave everything as it is; or (ii) re-write $r_y$ and $r_x$. There are correlation anomalies between rules $r_4$ and $r_9$, $r_5$ and $r_9$.

[3] In this case, each rule after $r_x$ will shift of one position.

It is important to note that Duplication, Contradiction and Correlation Anomalies are symmetric relations:

$$\mathscr{A}_{dupl}(r_x, r_y) \leftrightarrow \mathscr{A}_{dupl}(r_y, r_x) \qquad \mathscr{A}_{con}(r_x, r_y) \leftrightarrow \mathscr{A}_{con}(r_y, r_x) \qquad \mathscr{A}_{corr}(r_x, r_y) \leftrightarrow \mathscr{A}_{corr}(r_y, r_x)$$

Finally, the TABLE 1 summarizes the actions that an administrator can do in order to solve each intra-firewall anomaly. It is important to highlight that, in our view, except for the irrelevance anomalies, the other types of anomalies are not errors. In other words, each rule is individually correct and well written, but it is the interaction with other rules that may create anomalies.

| | | | |
|---|---|---|---|
| $\mathscr{A}_{dupl}(r_x, r_y)$ | $\mathscr{A}_{shaRed}(r_x, r_y)$ | $\mathscr{A}_{Unn}(r_x, r_y)$ | $\mathscr{A}_{irr}(r_x)$ |
| • remove $r_x$ | • remove $r_y$ | • remove $r_x$ | • remove $r_x$ |
| • remove $r_y$ | | | |
| $\mathscr{A}_{con}(r_x, r_y)$ | $\mathscr{A}_{shaConf}(r_x, r_y)$ | $\mathscr{A}_{cor}(r_x, r_y)$ | |
| • remove $r_y$ | • remove $r_y$ | •leave everything as it is | |
| • remove $r_x$ | • put $r_y$ just before $r_x$ | • re-write $r_y$ and $r_x$ | |

Table 1: Summary of fixing actions for each anomaly

# 4 Optimized Resolution

In this section we present the actual optimized resolution strategy, able to minimize the number of anomalies to be solved by the administrator. Specifically, our goal is to reduce the number of *conflict* anomalies that should be manually solved by administrators.

The evaluation of all the possible relations among a firewall rule set, and thus the identification of their anomalies, is a complex task that, given their formal definition presented before, can however be straightforwardly performed automatically by a software tool. On the other hand, the subsequent process of anomaly resolution is more difficult. As described in Section 2, in literature, some approaches that propose to resolve all the anomalies in a completely automatic way do exist [12]. However, this implies that the administrator has little to no control over the resolution process and particularly ambiguous and critical cases could be overlooked. Here we prefer a *semi-automatic* approach, that keeps the administrator always in control when possible ambiguities arise. More specifically, we propose a strategy to be followed that (i) avoids anomalies that are actually not relevant, (ii) does not introduce new anomalies and that (iii) quickly converges to a stable solution, without having to repeat the process multiple times.

Given the types and characteristics of the defined anomalies and the possible relations among the involved rules, it is possible to describe an optimized resolution strategy, that satisfies the requirements, as follows:

1. Consider firstly all the *Irrelevance* anomalies $\mathbb{A}_{irr}$. Clearly, the involved rules can be removed as they cannot affect the overall firewall behaviour. As a direct consequence, all anomalies (of different types) that include these same rules can be removed as well, *reducing* the set of anomalies to be considered;

$$\text{Remove } \forall r \in \mathbb{A}_{irr}:$$
$$\begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r \vee r_2 = r\}, \\ \alpha \in \{dupl, shaRed, con, shaConf, corr, unn\} \end{cases}$$

2. Then, consider all the *Duplication* anomalies $\mathbb{A}_{dupl}$. Each one of these anomalies involves two rules $r_x$ and $r_y$, where $r_y$ has a lower priority with respect to $r_x$ (that is $\pi(r_x) > \pi(r_y)$) and where the specified actions are the same. In this case, $r_y$ can always be safely removed, without affecting the firewall behaviour. Again, all the other anomalies that involve the removed rules, have to be cleared;

$$\text{Remove } \forall r_y | (r_x, r_y) \in \mathbb{A}_{dupl} : \begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r_y \vee r_2 = r_y\}, \\ \alpha \in \{dupl, shaRed, con, shaConf, corr, unn\} \end{cases}$$

3. Next consider the *Shadowing Redundancy* anomalies $\mathbb{A}_{shaRed}$, where, again, the actions of the involved rules are the same. Among all the anomalies of this type, we must consider firstly the anomalies rules where the corresponding $r_x$ has the highest priority. In fact, a shadowing rule can actually completely shadow pair of rules that are involved in other anomalies, thus "shadowing" the other anomalies themselves. For the same reason, among *Shadowing Redundancy* anomalies with the same $r_x$, the first to be considered are the ones with the highest priority $\pi(r_y)$. For each considered anomaly, it is then possible to remove the rule $r_y$ and propagate this removal by clearing the remaining anomaly set;

$$\text{Remove } \forall r_y | (r_x, r_y) \in \mathbb{A}_{shaRed} : \begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r_y \vee r_2 = r_y\}, \\ \alpha \in \{shaRed, con, shaConf, corr, unn\} \end{cases}$$

4. After this, we consider the *Contradiction* anomalies $\mathbb{A}_{con}$, where the rule conditions match exactly the same packets but with opposite actions. This case is similar to the *Duplication* one but does require an explicit evaluation from the administrator. He/she can decide either to (i) remove $r_x$ or (ii) remove $r_y$. Then the procedure continues in the usual way, by clearing all the affected anomalies.

(i) Remove $r_x, (r_x, r_y) \in \mathbb{A}_{con}$ :

$$\begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r_x \vee r_2 = r_x\}, \\ \alpha \in \{con, shaConf, corr, unn\} \end{cases}$$

(ii) Remove $r_y, (r_x, r_y) \in \mathbb{A}_{con}$ :

$$\begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r_y \vee r_2 = r_y\}, \\ \alpha \in \{con, shaConf, corr, unn\} \end{cases}$$

5. Next we consider every *Shadowing Conflict* anomaly $\mathbb{A}_{shaConf}$, where $r_x$ is shadowing $r_y$ and has an opposite action. Also in this case, the administrator has to evaluate the specific situation and decide how to act. Two possible decisions can be made: (i) remove $r_y$, (ii) move $r_y$ just before $r_x$ (meaning $\pi(r_y) > \pi(r_x)$). When (i) is applied the resolution process proceeds as usual, removing all the anomalies involving $r_y$. When (ii) is preferred, instead, the best course of actions is to choose firstly the anomaly with the highest priority (that is the highest $\pi(r_x)$) and move the corresponding $r_y$. Moving the rule $r_y$ to a higher priority place before $r_x$ means that the administrator has decided that, for all the packets matching the $C_y$ condition, the firewall must apply the $a_y$ action. This decision can be propagated and applied to solve other anomalies that involves $r_y$. For example, let's consider a rule $r_z = (C_z, a_x)$ for which $\mathscr{A}_{corr}(r_y, r_z)$ holds ($(r_y, r_z) \in \mathbb{A}_{corr}$). If $\pi(r_z) < \pi(r_x)$ we can confirm

the decision of the administrator and apply it also for those packets $p$ that match both $C_y$ and $C_z$. In fact, these packets are a subset of the packets that match $C_y$, and the administrator has already established the correct action to apply for $C_y$. For this reason, the correlation between $r_y$ and $r_z$ is not an anomaly anymore and can be removed. The same reasoning is valid if $\mathscr{A}_{shaConf}(r_y, r_z)$ holds.

(i) Remove $r_y, (r_x, r_y) \in \mathbb{A}_{shaConf}$ :

$$\begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | r_1 = r_x \vee r_2 = r_x\}, \\ \alpha \in \{shaConf, corr, unn\} \end{cases}$$

(ii) Move $r_y$ just before $r_x, (r_x, r_y) \in \mathbb{A}_{shaConf}$ :

$$\begin{cases} \mathbb{A}_\alpha = \mathbb{A}_\alpha \setminus \{(r_1, r_2) | \pi(r_z) < \pi(r_x)\}, \\ \alpha \in \{shaConf, corr, unn\} \end{cases}$$

6. *Correlation* anomalies are considered at this point. This kind of anomaly has to be solved by the administrator, who has two options: either to (i) leave everything as it is, or (ii) re-write $r_x$ and $r_y$. In the latter case, the anomaly detection and resolution process should be restarted in order to detect and solve the new anomalies potentially introduced by the administrator.

7. Finally, the resulting set of rules has to be evaluated again looking for *Unnecessary* anomalies only. It is worth noting, in fact, that at this point, after all the previous steps, the only kind of anomaly that can still be present is the *Unnecessary* one. Moreover, some new (*Unnecessary*) anomalies could have been introduced with respect to the initial set of rules, due to the shifts among other rules. These anomalies are dealt with in this last step as they can be easily solved by removing the $r_x$ rule (that is the rule with highest priority that is completely dominated by $r_y$).

At the end of this process, the initial set of rules have been reduced and all the anomalies solved.

# 5 Evaluation

In this section, we present the validation and implementation of the proposed approach.

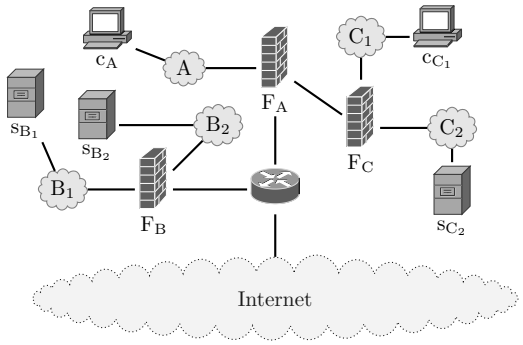
## 5.1 Validation

In order to validate the usefulness of our work, we have applied our optimized firewall anomaly resolution in a realistic scenario. Due to space limit and readability, we propose a network system which is small, but at the same time is able to show and stress the main aspects of the solution.

As shown in FIGURE 4, we have a network composed of three subnets $A$, $B$ and $C$. Each subnet is protected by a specific firewall $F_A$, $F_B$ and $F_C$. Moreover, we know some IP addresses, related to specific clients and servers of the network (i.e., $c_A, c_{C_1}, s_{B_1}, s_{B_2}, s_{C_2}$).

In this network scenario, we define six high-level policies:

- The subnet $A$ is allowed to reach the subnet $B$;
- The subnet $C_1$ and $C_2$ are not allowed to reach $B$;
- The network nodes inside the subnet $B$ are allowed to talk one to each other;
- Only $s_{B_2}$ and $c_A$ are allowed to talk with $s_{B_1}$
- Only $s_{C_2}$ and $s_{B_1}$ are allowed to talk with $s_{B_2}$
- Only a subset of network nodes (i.e, specified in other policies) is allowed to reach the subnet $B$.




LEGEND

router firewall filt. zone client server

VALUES

$A$: 10.1.0.0/24, $c_a$: 10.1.0.7
$B_1$: 10.2.0.0/24, $s_{B1}$: 10.2.0.5:80
$B_2$: 10.2.1.0/24, $s_{B2}$: 10.2.1.5:8080
$C_1$: 10.3.0.0/24, $c_{C1}$: 10.3.0.7
$C_2$: 10.3.1.0/24, $s_{C2}$:10.3.1.5:80 10.3.1.5:80,8080

Figure 4: The reference network

Wrong configuration

| | priority | IPsrc | Psrc | IPdst | Pdst | Proto | Action |
|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 10.1.0.0/24 | * | * | * | * | DENY |
| $r_2$ | 2 | 10.1.0.0/24 | * | * | * | TCP | ALLOW |
| $r_3$ | 3 | 10.1.0.0/24 | * | 10.3.0.0/24 | * | TCP | DENY |
| $r_4$ | 4 | 10.1.0.0/24 | * | 10.3.1.0/24 | * | TCP | DENY |
| $r_5$ | 5 | 10.1.0.7 | * | 10.2.0.5 | 80 | TCP | ALLOW |
| $r_6$ | 6 | 10.2.0.5 | 80 | 10.1.0.7 | * | TCP | ALLOW |
| $r_7$ | 7 | 10.2.0.5 | * | * | * | * | DENY |
| $r_8$ | 8 | * | * | 10.2.0.5 | * | * | DENY |
| $r_9$ | 9 | 10.2.1.5 | * | * | * | * | DENY |
| $r_{10}$ | 10 | * | * | 10.2.1.5 | * | * | DENY |
| $r_{11}$ | 11 | 10.3.1.5 | 80 | 10.2.1.5 | 8080 | TCP | ALLOW |
| $r_{12}$ | 12 | 10.2.1.5 | 8080 | 10.3.1.5 | 80 | TCP | ALLOW |
| $r_{13}$ | 13 | 10.2.0.0/24 | * | * | * | * | DENY |
| $r_{14}$ | 14 | * | * | 10.2.0.0/24 | * | * | DENY |
| $r_{15}$ | 15 | 10.2.1.0/24 | * | * | * | * | DENY |
| $r_{16}$ | 16 | * | * | 10.2.1.0/24 | * | * | DENY |
| $r_{17}$ | 17 | 10.2.0.5 | 80 | 10.2.1.5 | 8080 | TCP | ALLOW |
| $r_{18}$ | 18 | 10.2.1.5 | 8080 | 10.2.0.5 | 80 | TCP | ALLOW |
| $r_{19}$ | 19 | 10.2.0.0/24 | * | 10.2.1.0/24 | * | * | ALLOW |
| $r_{20}$ | 20 | 10.2.1.0/24 | * | 10.2.0.0/24 | * | * | ALLOW |
| $r_{21}$ | 21 | 10.3.0.0/16 | * | 10.2.0.0/16 | * | * | DENY |
| $r_{22}$ | 22 | 10.2.0.0/16 | * | 10.3.0.0/16 | * | * | DENY |
| default | $\infty$ | * | * | * | * | * | DENY |

Detected Anomalies

**Irrelevance**:
(1) $\mathcal{A}_{irr}(r_3)$
(2) $\mathcal{A}_{irr}(r_4)$

**Shad. Red.**:
(3) $\mathcal{A}_{shaRed}(r_1, r_3)$
(4) $\mathcal{A}_{shaRed}(r_1, r_4)$
(5) $\mathcal{A}_{shaRed}(r_2, r_5)$

**Correlation**:
(6) $\mathcal{A}_{corr}(r_7, r_{19})$
(7) $\mathcal{A}_{corr}(r_8, r_{20})$
(8) $\mathcal{A}_{corr}(r_9, r_{20})$
(9) $\mathcal{A}_{corr}(r_{10}, r_{19})$

**Unnecessary**:
(10) $\mathcal{A}_{unn}(r_7, r_{13})$
(11) $\mathcal{A}_{unn}(r_8, r_{14})$
(12) $\mathcal{A}_{unn}(r_{17}, r_{19})$
(13) $\mathcal{A}_{unn}(r_{18}, r_{20})$
(14) $\mathcal{A}_{unn}(r_{21}, r_{def})$
(15) $\mathcal{A}_{unn}(r_{22}, r_{def})$

**Shad. Conf.**:
(16) $\mathcal{A}_{shaConf}(r_1, r_2)$
(17) $\mathcal{A}_{shaConf}(r_1, r_5)$
(18) $\mathcal{A}_{shaConf}(r_2, r_3)$
(19) $\mathcal{A}_{shaConf}(r_2, r_4)$
(20) $\mathcal{A}_{shaConf}(r_7, r_{17})$
(21)$\mathcal{A}_{shaConf}(r_8, r_{18})$
(22) $\mathcal{A}_{shaConf}(r_9, r_{12})$
(23) $\mathcal{A}_{shaConf}(r_9, r_{18})$
(24) $\mathcal{A}_{shaConf}(r_{10}, r_{11})$
(25) $\mathcal{A}_{shaConf}(r_{10}, r_{17})$
(26) $\mathcal{A}_{shaConf}(r_{13}, r_{17})$
(27) $\mathcal{A}_{shaConf}(r_{13}, r_{19})$
(28) $\mathcal{A}_{shaConf}(r_{14}, r_{18})$
(29) $\mathcal{A}_{shaConf}(r_{14}, r_{20})$
(30) $\mathcal{A}_{shaConf}(r_{15}, r_{18})$
(31) $\mathcal{A}_{shaConf}(r_{15}, r_{20})$
(32) $\mathcal{A}_{shaConf}(r_{16}, r_{17})$
(33) $\mathcal{A}_{shaConf}(r_{16}, r_{19})$

Figure 5: The $F_B$ configuration and its intra-firewall anomalies

Resolved Anomalies

**Irrelevance**:
(1) $\mathcal{A}_{irr}(r_3)$
(2) $\mathcal{A}_{irr}(r_4)$

**Shad. Red.**:
(5) $\mathcal{A}_{shaRed}(r_2, r_5)$

**Shad. Conf.**:
(16) $\mathcal{A}_{shaConf}(r_1, r_2)$
(20) $\mathcal{A}_{shaConf}(r_7, r_{17})$
(21)$\mathcal{A}_{shaConf}(r_8, r_{18})$
(22) $\mathcal{A}_{shaConf}(r_9, r_{12})$
(24) $\mathcal{A}_{shaConf}(r_{10}, r_{11})$
(27) $\mathcal{A}_{shaConf}(r_{13}, r_{19})$
(29) $\mathcal{A}_{shaConf}(r_{14}, r_{20})$
(31) $\mathcal{A}_{shaConf}(r_{15}, r_{20})$

**Correlation**:
(6) $\mathcal{A}_{corr}(r_7, r_{19})$
(7) $\mathcal{A}_{corr}(r_8, r_{20})$
(8) $\mathcal{A}_{corr}(r_9, r_{20})$
(9) $\mathcal{A}_{corr}(r_{10}, r_{19})$

**Unnecessary**:
(14) $\mathcal{A}_{unn}(r_{21}, r_{def})$
(15) $\mathcal{A}_{unn}(r_{22}, r_{def})$
(34) $\mathcal{A}_{unn}(r_1, r_{def})$
(35) $\mathcal{A}_{unn}(r_{13}, r_{def})$
(36) $\mathcal{A}_{unn}(r_{14}, r_{def})$
(37) $\mathcal{A}_{unn}(r_{15}, r_{def})$
(38) $\mathcal{A}_{unn}(r_{16}, r_{def})$

Novel configuration

| | priority | IPsrc | Psrc | IPdst | Pdst | Proto | Action |
|---|---|---|---|---|---|---|---|
| $r_2$ | 1 | 10.1.0.0/24 | * | * | * | TCP | ALLOW |
| $r_6$ | 2 | 10.2.0.5 | 80 | 10.1.0.7 | * | TCP | ALLOW |
| $r_{11}$ | 3 | 10.3.1.5 | 80 | 10.2.1.5 | 8080 | TCP | ALLOW |
| $r_{12}$ | 4 | 10.2.1.5 | 8080 | 10.3.1.5 | 80 | TCP | ALLOW |
| $r_{17}$ | 5 | 10.2.0.5 | 80 | 10.2.1.5 | 8080 | TCP | ALLOW |
| $r_{18}$ | 6 | 10.2.1.5 | 8080 | 10.2.0.5 | 80 | TCP | ALLOW |
| $r_7$ | 7 | 10.2.0.5 | * | * | * | * | DENY |
| $r_8$ | 8 | * | * | 10.2.0.5 | * | * | DENY |
| $r_9$ | 9 | 10.2.1.5 | * | * | * | * | DENY |
| $r_{10}$ | 10 | * | * | 10.2.1.5 | * | * | DENY |
| $r_{19}$ | 11 | 10.2.0.0/24 | * | 10.2.1.0/24 | * | * | ALLOW |
| $r_{20}$ | 12 | 10.2.1.0/24 | * | 10.2.0.0/24 | * | * | ALLOW |
| default | $\infty$ | * | * | * | * | * | DENY |

Figure 6: The resolved anomalies and the novel $F_B$ configuration

In this example, the network security administrator has followed these high-level policies to implement the configurations of all the firewalls, including the specific configuration for firewall $F_B$, shown in FIGURE 5. However, this initial configuration has some flaws. Specifically, there are 33 anomalies: 2 Irrelevance, 3 Shadowing Redundancy, 18 Shadowing Conflict, 4 Correlation and 6 Unnecessary. These anomalies are listed and numbered in FIGURE 5.

In this scenario we applied the proposed algorithm described in the previous Section, following these steps:

1. we solved the Irrelevance anomaly (1) by removing rule $r_3$.
   Consequently anomalies (3) and (18) disappear;

2. we solved the Irrelevance anomaly (2) by removing rule $r_4$.
   Consequently anomalies (4) and (19) disappear;

3. we solved Shadowing Redundancy anomaly (5) by removing rule $r_5$.
   Consequently anomaly (5) disappears;

4. we solved the Shadowing Conflict anomaly (16) by moving rule $r_2$ before $r_1$;

5. we solved the Shadowing Conflict anomaly (20) by moving rule $r_{17}$ just before $r_7$.
   Consequently anomalies (12), (25), (26) and (32) disappear;

6. we solved the Shadowing Conflict anomaly (21) by moving rule $r_{18}$ just before $r_8$.
   Consequently anomalies (13), (23), (28) and (30) disappear;

7. we solved the Shadowing Conflict anomaly (22) by moving rule $r_{12}$ just before $r_9$;

8. we solved the Shadowing Conflict anomaly (24) by moving rule $r_{11}$ just before $r_{10}$;

9. we solved the Shadowing Conflict anomaly (27) by moving rule $r_{19}$ just before $r_{13}$.
   Consequently anomalies (10) and (33) disappear;

10. we solved the Shadowing Conflict anomaly (29) by moving rule $r_{20}$ just before $r_{14}$.
    Consequently anomalies (11) and (31) disappear;

11. we marked the Correlation anomalies (6), (7), (8), (9), as solved,
    assuming the administrator accepts them;

12. we solved the Unnecessary anomalies (14) and (15) respectively by removing rules $r_{21}$ and $r_{22}$;

13. finally, we solved the newly introduced Unnecessary anomalies (34),(35), (36), (37) and (38)
    respectively by removing rules $r_1$, $r_{13}$, $r_{14}$, $r_{15}$ and $r_{16}$.

As shown in FIGURE 6, using our optimized resolution, we reduced the number of rules to 12 by solving only 22 anomalies, 16 of which are coming from the original flawed configuration and 5 are new Unnecessary anomalies introduced by the resolution process. It is worth remarking that this type of anomaly is easily solved by removing one rule, thus reducing the set of rules without requiring

administrator's actions.

It is important to note that the number of anomalies that are automatically removed depends on the specific scenario. However, it is evident that the proposed approach significantly reduces the administrator's effort.

### 5.2 Implementation

We implemented the proposed methodology and techniques in a prototype software tool. This prototype has been written in Java for portability and is composed by two modules.

The first module reads the description of a network topology and the configuration of the included firewalls. Then, it performs a first analysis to collect all the anomalies among firewall rules, as defined in previous sections.

The second module evaluates the complete set of anomalies and follows the optimized resolution strategy to reduce the total number of anomalies to solve. When an administator decision is requested (points 4 and 5 in Section 4), a random action is selected. When *correlation* anomalies are evaluated (point 5), it is assumed that the administator accepts the current situation (and leave everything as it is).

We tested this implementation on the example presented in previous sections and we built several other test cases based on larger synthetic networks. In each test we randomly changed the order of the rules to generate different initial anomaly sets.

These tests were performed to assess the feasibility of the methodology, even in a prototype version of the tool, in its two main aspects: to find the anomalies and to suggest the optimized resolution strategy.

As preliminary results, the conducted tests confirmed the feasibility of the approach applied to realistic cases. In all the experiments the number of total anomalies to be considered has been reduced with respect to the initial set, confirming the usefulness of the approach.

## 6 Conclusion and Future Work

In this work, we presented a formal approach to optimize the resolution of firewall anomalies. Specifically, our proposed approach is able to reduce the set of anomalies that an administrator has to manually solve in order to remove all the anomalies among firewalls rules. Moreover, we propose a novel rule relations model and intra-firewall anomaly classification. Finally, the feasibility of the proposed approach was evaluated considering a realistic network scenario analyzed by a Java-based prototype software implementing the presented methodology.

As possible future work, we plan to extend the expressiveness and capabilities of our model to make it able to solve problems related to attribute-based access control models, strictly related to firewall behaviours, and to perform some empirical study with network security administrator, in order to improve the usability of our tool and approach.

## Acknowledgments

This work was supported in part by the European Commission, under Grant Agreement no. 830929.

## References

[1] E. Al-Shaer, H. Hamed, R. Boutaba, and M. Hasan. Conflict classification and analysis of distributed firewall policies. *IEEE Journal on Selected Areas in Communications*, 23(10):2069–2084, October 2005.

[2] M. Baldi and G. Marchetto. Pipeline forwarding of packets based on a low-accuracy network-distributed common time reference. *IEEE/ACM Transactions on Networking*, 17(6):1936–1949, December 2009.

[3] M. Baldi, G. Marchetto, F. Risso, G. Galante, R. Scopigno, and F. Stirano. Time driven priority router implementation and first experiments. In *Proc. of the 2006 IEEE International Conference on Communications (ICC'06), Istanbul, Turkey*, pages 897–902. IEEE, June 2006.

[4] C. Basile, D. Canavese, A. Lioy, and F. Valenza. Inter-technology conflict analysis for communication protection policies. In *Proc. of the 2014 International Conference on Risks and Security of Internet and Systems (CRiSIS'14), Trento, Italy*, volume 8924 of *Lecture Notes in Computer Science*, pages 148–163. Springer, Cham, August 2015.

[5] C. Basile, D. Canavese, C. Pitscheider, A. Lioy, and F. Valenza. Assessing network authorization policies via reachability analysis. *Computer and Electrical Engineering*, 64(C):110–131, November 2017.

[6] C. Basile, F. Valenza, A. Lioy, D. R. Lopez, and A. Pastor Perales. Adding support for automatic enforcement of security policies in nfv networks. *IEEE/ACM Transactions on Networking*, 27(2):707–720, April 2019.

[7] N. Basumatary and S. M. Hazarika. Model checking a firewall for anomalies. In *Proc. of the first International Conference on Emerging Trends and Applications in Computer Science (ICETACS'13), Shillong, India*, pages 92–96. IEEE, September 2013.

[8] R. Boutaba and I. Aib. Policy-based management: A historical perspective. *Journal of Network and Systems Management*, 15(4):447–480, November 2007.

[9] M. Cheminod, L. Durante, L. Seno, F. Valenza, and A. Valenzano. A comprehensive approach to the automatic refinement and verification of access control policies. *Computers and Security*, 80:186 – 199, January 2019.

[10] M. Cheminod, L. Durante, L. Seno, F. Valenza, A. Valenzano, and C. Zunino. Leveraging sdn to improve security in industrial networks. In *Proc. of the 2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS'17), Trondheim, Norway*, pages 1–7. IEEE, May 2017.

[11] L. Durante, L. Seno, F. Valenza, and A. Valenzano. A model for the analysis of security policies in service function chains. In *Proc. of the 2017 IEEE Conference on Network Softwarization (NetSoft'17), Bologna, Italy*, pages 1–6. IEEE, July 2017.

[12] H. Hu, G. Ahn, and K. Kulkarni. Detecting and resolving firewall policy anomalies. *IEEE Transactions on Dependable and Secure Computing*, 9(3):318–331, May 2012.

[13] H. Hu, G.-J. Ahn, and K. Kulkarni. Fame: A firewall anomaly management environment. In *Proc. of the 3rd ACM Workshop on Assurable and Usable Security Configuration (SafeConfig'10), Chicago, Illinois, USA*, page 17–26. ACM, October 2010.

[14] R. Hunt. Internet/intranet firewall security—policy, architecture and transaction services. *Computer Communications*, 21(13):1107 – 1123, September 1998.

[15] A. A. Jabal, M. Davari, E. Bertino, C. Makaya, S. Calo, D. Verma, A. Russo, and C. Williams. Methods and tools for policy analysis. *ACM Computing Surveys*, 51(6):121:1–121:35, February 2019.

[16] W. John, G. Marchetto, F. Nemeth, P. Skoldstrom, R. Steinert, C. Meirosu, I. Papafili, and K. Pentikousis. Service provider devops. *IEEE Communications Magazine*, 55(1):204–211, January 2017.

[17] A. Khoumsi, W. Krombi, and M. Erradi. A formal approach to verify completeness and detect anomalies in firewall security policies. In *Proc. of the 7th International Symposium on Foundations and Practice of Security (FPS'14), Montreal, Quebec, Canada*, volume 8930 of *Lecture Notes in Computer Science*, pages 221–236, November 2014.

[18] W. Krombi, M. Erradi, and A. Khoumsi. Automata-Based Approach to Design and Analyze Security Policies. In *Proc. of the 12th International Conference on Privacy, Security and Trust (PST'14), Toronto, Ontario, Canada*, pages 306–313. IEEE, July 2014.

[19] N. Lehmann, R. Schwarz, and J. Keller. FIRECROCODILE: A Checker for Static Firewall Configurations. In *Proc. of the International Conference on Security & Management (SAM'06), Las Vegas, Nevada, USA*, pages 193–199. CSREA Press, June 2006.

[20] A. Liu and M. Gouda. Structured firewall design. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, (4):1106–1120, March 2007.

[21] A. X. Liu and M. G. Gouda. Complete Redundancy Detection in Firewalls. In *Proc. of the 19th Annual IFIP*

*WG 11.3 Working Conference on Data and Applications Security (DBsec'05), Storrs, Connecticut, USA*, volume 3654 of *Lecture Notes in Computer Science*, pages 193–206. Springer, Berlin, Heidelberg, August 2005.

[22] E. C. Lupu and M. Sloman. Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering*, 25(6):852–869, November 1999.

[23] F. Valenza, C. Basile, D. Canavese, and A. Lioy. Classification and analysis of communication protection policy anomalies. *IEEE/ACM Transactions on Networking*, 25(5):2601–2614, October 2017.

[24] F. Valenza, S. Spinoso, C. Basile, R. Sisto, and A. Lioy. A formal model of network policy analysis. In *Proc. of the 2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI'15), Turin, Italy*, pages 516–522. IEEE, September 2015.

[25] Verizon. Data Breach Investigations Report. `https://enterprise.verizon.com/en-au/resources/reports/dbir/` [Online; accessed on February 5, 2020], 2019.

[26] P. Verma and A. Prakash. FACE: A Firewall Analysis and Configuration Engine. In *Proc. of the 2005 Symposium on Applications and the Internet (SAINT'05), Trento, Italy*, pages 74–81. IEEE, February 2005.

[27] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, and S. Waldbusser. Terminology for Policy-Based Management. IETF RFC 3198, November 2001. `http://www.rfc-editor.org/rfc/rfc3198.txt` [Online; accessed on February 5, 2020].

[28] L. Yuan, H. Chen, J. Mai, and C.-n. Chuah. FIREMAN: A Toolkit for FIREwall Modeling and ANalysis. In *Proc. of the 2006 IEEE Symposium on Security and Privacy (S&P'06), Oakland, California, USA*, pages 199–213. IEEE, May 2006.

---

## Author Biography

**Fulvio Valenza** received the M.Sc. degree (summa cum laude) and the Ph.D. degree (summa cum laude) in computer engineering from the Politecnico di Torino, Torino, Italy, in 2013 and 2017, respectively, where he is currently a Researcher. His research activity focuses on network security policies, orchestration and management of network security functions in SDN/NFV-based networks, and threat modeling.

**Manuel Cheminod** received the M.S. in 2005 and the Ph.D in Computer Engineering in 2010, both from the Politecnico di Torino, Italy. He his a Researcher with the Institute of Electronics, Computer, and Telecommunication Engineering of the National Resarch Council of Italy, in Torino. His research interests include formal methods for the verification of security properties in industrial systems and the exploitation of SDN/NFV paradigms to improve the security level of the same critical systems.