Inter-technology conflict analysis for communication protection policies

Inter-Technology Conflict Analysis for Communication Protection Policies

Cataldo Basile, Daniele Canavese, Antonio Lioy, and Fulvio Valenza

Politecnico di Torino, Dip. di Automatica e Informatica, Italy
{cataldo.basile,daniele.canavese,antonio.lioy,fulvio.valenza}@polito.it

Abstract. Usually network administrators implement a protection policy by refining a set of (abstract) communication security requirements into configuration settings for the security controls that will provide the required protection. The refinement consists in evaluating the available technologies that can enforce the policy at node and network level, selecting the most suitable ones, and possibly making fine adjustments, like aggregating several individual channels into a single tunnel. The refinement process is a sensitive task which can lead to incorrect or sub-optimal implementations, that in turn affect the overall security, decrease the network throughput and increase the maintenance costs. In literature, several techniques exist that can be used to identify anomalies (i.e. potential incompatibilities and redundancies among policy implementations. However, these techniques usually focus only on a single security technology (e.g. IPsec) and overlook the effects of multiple overlapping protection techniques. This paper presents a novel classification of communication protection policy anomalies and a formal model which is able to detect anomalies among policy implementations relying on technologies that work at different network layers. The result of our analysis allows administrators to have a precise insight on the various alternative implementations, their relations and the possibility of resolving anomalies, thus increasing the overall security and performance of a network.

1 Introduction

Nowadays, computers have a pervasive presence in all our daily activities. The current technological trend is to reduce the human intervention and provide human beings with precise information that can be used for decision-making purposes. This is particularly important in areas where human lives, high economic costs and security in general are at stake. The final target is to lessen the human fallibility.

Protecting a networked IT infrastructure, guaranteeing user privacy, and securing communications are important facets of this technological evolution. For a human being is very difficult (if not actually impossible) to envision the whole configuration of large networked systems and implement it without errors and with an adequate amount of protection. As several studies have proven, the human factor is the main cause of misconfigurations [1,2]. To this purpose, a number of (semi-)automatic tools and techniques have been proposed. For instance,

the policy-based network management paradigm proposes to define the security requirements by means of a set of business-level "statements" (the policy), that are later manually or semi-automatically refined into low-level configurations for the available security controls.

Communication protection policies are used to specify how to protect the network communications. Their correct deployment is crucial in several areas, such as protection of intellectual properties, and confidentiality of financial or corporate data (like credit card numbers). The specification of communication protection policies simply requires the definition of the communication end-points to protect (with minimal or no clues about the path in between), seldom the security properties to ensure (e.g. confidentiality and data integrity), and, seldom if ever, hints about the technology to adopt. Therefore, the refinement of communication protection policies is challenging since an administrator, or a tool mimicking his behaviour, must automatically infer and choose several technical details among several alternatives, such as the security protocol (e.g. SSL/TLS, SSH or S-FTP), the cipher-suite, the timeouts, and so on. To make the refinement easier to implement and manage, the transformation towards the low-level configurations is frequently split in several steps which make use of a series of intermediate representations of the policies that are enriched with new technical data at each step.

For instance, by adding to communication protection policies the type of protection (message or channel), the ISO/OSI layer where the protection must be implemented, the technology to use (e.g. IPsec VPN or TLS) and the security properties to enforce (such as header integrity, payload integrity, confidentiality), we obtain a middle-level policy representation. In this paper we will call this representation a *policy implementation*, or *PI*. Refining a set of PIs can lead to incompatibilities and redundancies, named *anomalies*, that cannot be detected at the upper layers, but are best noticed at the policy implementation level. Anomalies appear during the refinement due to several factors. First, the introduction of new technological parameters can produce overlapping or redundant PIs. Secondly, since the refinement is usually performed iteratively, one communication protection policy at the time, interactions between policy implementations are not known a-priori and can only be resolved a-posteriori.

In this paper, we propose a novel approach which is able to detect a number of anomalies between PIs taking into account the interactions between several technologies and security properties. Our model uses a set of FOL (first order logic) axioms which guarantee accurate results and performances. In literature a number of works on policy and configuration anomaly detection exists. Several notable works in this area are due to Al-Shaer, which proposed a model and a taxonomy of conflicts on low-level configurations for communication protections, with a particular focus on the IPsec protocol [3,4]. Another interesting work is by Li et al. [5]: it classifies the IPsec rules in access control lists and encryption lists. Most of the literature, however, focuses only on a single communication protection technology, thus lacking a way to classify and detect inter-technologies conflicts.

Our contribution to the state-of-the-art is three-fold:

- our model allows the detection of a number of anomalies arising from the interactions between various protocols (e.g. TLS and SSH), security properties and communication scenarios such as end-to-end connections, VPNs and remote access communications (see RFC-3457 [6]). We take into account both communication end-points (i.e. source and destination), but also tunnel terminators/gateways for a more accurate detection. To the best of our knowledge this is the first work that detects and classifies communication inter-technology policy anomalies;
- our model allows the detection of anomalies at different ISO/OSI layers, i.e. conflicts involving IP addresses, ports and URIs (e.g. for web services). Our approach internally represents every network device as a tree containing various "entities", able to establish or terminate secure communications, which live at different ISO/OSI levels. Our hierarchical view of networks and network nodes improves on existing works, which often only rely on a flat IP address-based representation of an IT infrastructure;
- we provides the administrators an effective reporting of the anomalies. Having built our model on a FOL family, we can use all its well-known equivalences and logical properties. For instance, we use an easy-to-read multigraph representation to show the administrators the relationships between the anomalous implementations.

This paper is structured as follows. Section 2 provides an example which we use to introduce our model. Section 3 detail our mathematical model, the anomaly classification, and the axioms for detecting them. Section 4 describes our implementation using ontological techniques and the graphical notations for reporting the anomalies. Finally, Section 5 and 6 discuss the related works and the conclusions on our approach.

2 A motivating example

In this section we present an example which will be used to informally introduce the concepts of our model. We will use as a reference the simplified network scenario depicted in Fig. 1.

Fig. 1 shows two subnets connected through an insecure area. The subnet $\mathcal{C}$, on the left, consists of a number of administrative assistants have their own workstations and use their clients to connect to the subnet $\mathcal{S}$, on the right, where several company servers are deployed. The server S_1 provides two services, a web service, where the administrative assistants in $\mathcal{C}$ can access a number of administrative functions, and the company database, containing the data about the employed, which the administrative assistants can query via a stand-alone client. Fig. 2 presents the 'internals' of the S_1 server using a tree-like notation that emphasizes the different ISO/OSI levels involved. The node labeled 'S_1' aggregates all the lowest levels and plays the role of a placeholder for all the communications to and from S_1. The node 's_1^3', lying below, represents the network

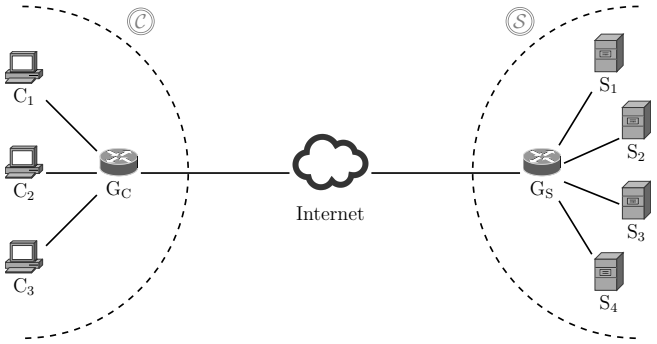

Fig. 1: A simplified network scenario.

layer (ISO/OSI level 3), where techniques are available to enforce network level channel protection (e.g. IPsec). This node forks in two branches, corresponding to two open ports, the port 3306 where the company database is waiting for requests, and the port 8080 where the administrative web service is available. These nodes are respectively labeled 's^4_{1a}' and 's^4_{1b}' and are both at the transport layer (ISO/OSI level 4), where channel techniques are available to enforce the traffic protection (e.g. TLS/SSL and SSH)[1]. Finally, we have the node labeled with 's^7_{1b}' at the application level (ISO/OSI level 7), where a message protection protocol can be chosen to secure the communication (e.g. WS-Security).

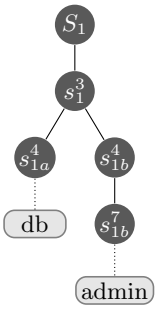

Fig. 2: Graphical representation of the S_1 node structure.

Let us consider the case where two high-level policies need to be refined and implemented. The policy p_1 states "users with the administrative assistant role must securely access the administration web service", and p_2 states "users with

[1] It is possible to debate about TLS and SSH being protocols that work at transport or session layer and if SSH is actually a general purpose channel protection protocol. We avoid to enter this discussion as both techniques, from our (practical) point of view, can be used to protect all the communications regarding a given port.

the administrative assistant role must securely access the company database". Even in this minimalistic network scenario, there are many different ways to enforce these policies. For instance, by just considering the administrative assistant working on the client C_1, there are the following alternative implementations for p_1:

- $i_{1,1}$ establishes a secure end-to-end channel between C_1 and s_1^3 at level 3 of the ISO/OSI stack with IPsec;
- $i_{1,2}$ creates a secure end-to-end channel between C_1 and s_{1b}^4 at level 4 with TLS;
- $i_{1,3}$ applies a message protection technique at level 7 to secure the messages exchanged between C_1 and s_{1b}^7 (e.g. by using WS-Security);
- $i_{1,4}$ uses a secure tunnel between G_C and G_S at layer 3 of the ISO/OSI stack, with IPsec in tunnel model, to protect the communications between C_1 and S_1;

For the policy p_2, instead, we have the following PIs:

- $i_{2,1}$ establishes a secure end-to-end channel between C_1 and s_1^3 at layer 3 of the ISO/OSI stack with IPsec;
- $i_{2,2}$ makes use of a secure end-to-end channel between C_1 and s_{1a} at level 4 with SSH;
- $i_{2,3}$ uses a secure tunnel between G_C and G_S at layer 3 of the ISO/OSI stack, with IPsec in tunnel model, to protect the communications between C_1 and S_1.

Furthermore, both p_1 and p_2 can be simultaneously enforced by establishing a single secure tunnel between G_C and G_S that protects the entire communications between the subnets $\mathcal{C}$ and $\mathcal{S}$. We will name this policy implementation $i_{\mathcal{C},\mathcal{S}}$.

These PIs present some peculiar properties. For instance, $i_{1,1}$ and $i_{2,1}$ require the enforcement of the same channel, that can be used to protect the communication towards both the services of S_1. In this case, $i_{1,1}$ and $i_{2,1}$ are *equivalent* implementations, the simplest anomaly type.

On the other hand, the implementation $i_{1,1}$ "protects more than" $i_{1,2}$, $i_{1,3}$ and $i_{2,2}$. Given the network stack, if the communications between two nodes are protected at layer 3, also all the layer 4 communications are protected. In this case, we have another type of anomaly, the *inclusion*. Inclusion anomalies have however a much broader spectrum. For example, we have also an inclusion if two PIs share the same end-points, but one implementation requires more security properties than the other (e.g. confidentiality and integrity instead of confidentiality only).

Another kind of anomaly we classified is the *affinity*, which indicates that two PIs share some common aspects, but none of the involved implementation includes the other. For instance, we have an affinity anomaly if two PIs have the two sources/destinations on the same node and/or if they impose 'disjoint' security properties (e.g., confidentiality only vs. data integrity only), like for $i_{1,2}$ and $i_{2,2}$. It is worth noting that affine PIs have an interesting property. There exists

another "more general" PI that can substitute both the affine implementation, e.g., in the previous case $i_{1,1}$ can substitute both $i_{1,2}$ and $i_{2,2}$.

Another anomaly that we have identified is the *alternative* anomaly, that arises when two PIs have the same end-points but the in-between path is different. For instance, $i_{\mathcal{C},\mathcal{S}}$ is an alternative to both $i_{1,1}$ and $i_{2,1}$[2].

In addition to the previously mentioned anomalies, involving PI pairs, we categorize another set of anomalies concerning only one policy implementation[3]:

- a PI is *inadequate*, if the secure communication does respects a set of minimum requirements defined by the administrators. For instance, if the minimum requirement is 'all the data transfers must be encrypted', the implementation $i_{1,4}$ is inadequate since the traffic inside the subnets $\mathcal{C}$ and $\mathcal{S}$ is sent in the clear;
- a PI is *filtered* if a secure communication is truncated by some filtering device[4];
- a PI is *irrelevant* if its removal does not alter the semantic of the network. For instance, a policy implementation with the same source and destination is irrelevant.

3 Policy implementation analysis and resolution

In this section, we formally define our mathematical model. First, we give the definition of policy implementation, then we introduce our taxonomy of conflicts and finally we discuss the logical axioms needed to identify and resolve such anomalies.

3.1 Formal definition of policy implementation

In our model, a policy implementation i is:

$$i = (s, d, t, c^h, c^p, c^c, G)$$

The symbols s and d respectively represent the PI source and destination. The symbol t specifies the adopted technology. In our analysis we will consider five technologies: IPsec, TLS, SSH, WS-Security and NULL. The NULL technology indicates that a communication should be created without any kind of protection. Our model is however extensible to other technologies, especially at application layer. The fields c^h, c^p and c^c are three Boolean values that indicate a

[2] To be more precise, from the security point of view, $i_{\mathcal{C},\mathcal{S}}$ can be considered equivalent to $i_{1,1}$ and $i_{2,1}$ only if both the subnets are considered trusted.

[3] An well designed automatic refinement would never introduce these anomalies, but detecting them is nevertheless useful in case of manual refinement.

[4] Technically a filtered PI is an anomaly between a communication protection PI and a filtering PI, but in this paper we are only interested in communication protection policies.

required security property. They respectively denote the header integrity, payload integrity and confidentiality. If the chosen technology is NULL, obviously all these properties are false. The latest symbol $G = (g_1, \ldots, g_n)$ is an ordered list of gateway nodes. This information is particularly useful when analyzing site-to-site or remote access communications. In an end-to-end connection obviously $G = \varnothing$.

For an accurate detection, we need to precisely identify the layer in the ISO/OSI stack where a communication starts and terminates. To this purpose, we defined a hierarchical structure (a tree) that describes the points where the secure communications can be established for each network node (see Fig. 2). The root node represents the whole network node, while the other tree nodes model the available 'connection points' in the TCP/IP stack, that is, network, transport and application layers. These tree nodes may optionally be associated to IP addresses, ports and URIs.

We defined a set of relationships between the 'connection points', that are the network elements that play the role of source and destinations in a PI. Given two elements e_1 and e_2, we have:

- *equivalence* between e_1 and e_2, if they are exactly the same element. We denote this condition with $e_1 = e_2$;
- *dominance* of e_1 over e_2, if all the communications starting from/arriving to e_2 pass through e_1. We denote this condition with $e_1 \succ e_2$. This concept is useful when dealing with protocols working at different layers. For instance, an entity at layer 3 dominates all the transport and application nodes beneath it;
- *disjointness* between e_1 and e_2, if e_1 and e_2 belong to different network nodes. We denote this condition with $e_1 \perp e_2$. Note that if e_1 and e_2 are on the same device we will write $e_1 \not\perp e_2$ (that is they are not disjoint).

In a similar way to the network elements, we can define a number of relationships amongst the technologies. Given two technologies t_1 and t_2, we have:

- *equivalence* between t_1 and t_2, if they are exactly the same technology. We denote this condition with $t_1 = t_2$;
- *dominance* of t_1 over t_2, if the ISO/OSI layer of t_1 is strictly less than the layer of t_2. We denote this condition with $t_1 \succ t_2$. Obviously, NULL is dominated by all the other technologies;
- *disjointness* between t_1 and t_2, if t_1 and t_2 work at the same ISO/OSI layer and $t_1 \neq t_2$. We denote this condition with $t_1 \perp t_2$.

For the subset of technologies that we considered in this paper, the following relations hold:

$$\text{IPsec} \succ \text{TLS} \succ \text{WS-Security} \succ \text{NULL}$$
$$\text{IPsec} \succ \text{SSH} \succ \text{WS-Security} \succ \text{NULL}$$
$$\text{SSH} \perp \text{TLS}$$

Given two properties c_1^x and c_2^x (in the same field $x = \{h, p, c\}$), we have:

– *equivalence* between c_1^x and c_2^x, if they have the same value. We denote this condition with $c_1^x = c_2^x$;
– *dominance* of c_1^x over c_2^x, if $c_1^x = true$ and $c_2^x = false$. We denote this condition with $c_1^x \succ c_2^x$.

Finally, given two lists of gateway nodes G_1 and G_2, we have:

– *equivalence* between G_1 and G_2, if they have the same gateways in the same order. We denote this condition with $G_1 = G_2$;
– *disjointness* between G_1 and G_2, if they have at least one ordered couple of gateways not in common. For instance (gw_1, gw_2, gw_3) and (gw_1, gw_3, gw_2) are disjoint. We denote this condition with $G_1 \perp G_2$.

3.2 Algebraic representation of anomalies

Having now at our disposal a formal definition of a policy implementation, we can formalize, from an algebraic point of view, the anomalies that may occur between two PIs and the anomalies in the specification of a single PI.

Fig. 3 presents the proposed taxonomy of the anomalies. In the following sections we will give both the axioms to detect such conflicts and a relative resolution strategy.

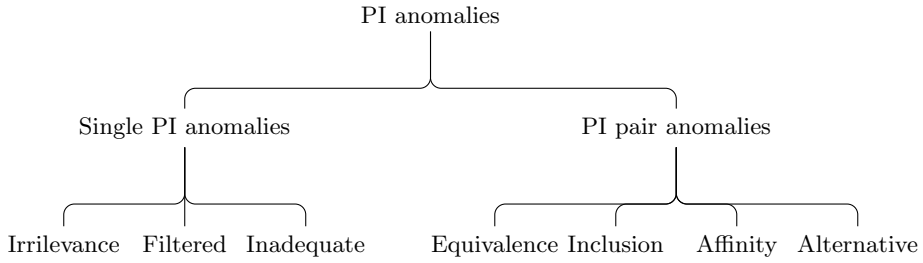

Fig. 3: The proposed taxonomy of anomalies.

Single PI anomalies

A single PI anomaly occurs within a policy implementation itself. Given a policy implementation $i = (s, d, t, c^h, c^p, c^c, G)$, we have identified three kind of anomalies in this category.

A PI is *irrelevant* when the network behaviour remains unaltered if the PI is removed. These anomalies can be inferred using the axiom:

$$s \not\perp d$$

that identifies implementations such that the source and the destination lays on the same node, thus creating a sort of loop. The proposed resolution is to delete i.

A PI is *filtered* when there exists at least a filtering device in the network path that contains a filtering rule f that discards the traffic related to the policy implementation. Given an oracle-like function $\mathcal{F}(i)$, which returns true if the traffic related to i is discarded and false otherwise, we can identify this anomalies using the formula:

$$\mathcal{F}(i) = true$$

In practice, the output of this oracle-like function can be populated by means of a network reachability analysis [7]. This anomaly is the evidence of a number of severe errors in the policy definitions that can have dangerous repercussions on the security and connectivity of the network. In order to remove the anomaly, the administrator can choose to remove the PI or the filtering rule f.

A PI is *inadequate* when its security properties establish a security level lower than an acceptable threshold. Given an oracle-like function $\mathcal{C}(i)$ which returns true when the policy implementation i is considered protected and false otherwise, we can detect these anomalies with the trivial equation:

$$\mathcal{C}(i) = true$$

The oracle is a function that checks the security requirements defined a priori by the network administrators. For example an network administrator could define an oracle establishing that all the communications coming from and destined to the Internet must be confidential, while the internal communication could respect just the data integrity property.

The simplest way to implement the function $\mathcal{C}$ in practice is to define a triple $(\tilde{c}^h, \tilde{c}^p, \tilde{c}^c)$ defining the minimum security levels, so that:

$$\mathcal{C}(i) = \begin{cases} true & \text{if } c^h \succeq \tilde{c}^h \wedge c^p \succeq \tilde{c}^p \wedge c^c \succeq \tilde{c}^c \\ false & \text{otherwise} \end{cases}$$

To solve these anomalies the security properties of the policy implementation must be modified accordingly, for instance by setting to true the properties than are required to be true by the triple $(\tilde{c}^h, \tilde{c}^p, \tilde{c}^c)$.

PI pair anomalies

Given two PIs $i_1 = (s_1, d_1, t_1, c_1^h, c_1^p, c_1^c, G_1)$ and $i_2 = (s_2, d_2, t_2, c_2^h, c_2^p, c_2^c, G_2)$, our model allows the detection of a number of anomalies.

Two policy implementations i_1 and i_2 are *equivalent*, and we will write $i_1 = i_2$, if they have the same values for all their tuple fields, that is they are equivalent if:

$$s_1 = s_2 \wedge d_1 = d_2 \wedge t_1 = t_2 \wedge c_1^h = c_2^h \wedge c_1^p = c_2^p \wedge c_1^c = c_2^c \wedge G_1 = G_2$$

These anomalies are trivially resolved by removing one of the two PIs.

The policy implementation i_1 *includes* (or dominates) the policy implementation i_2, and we will write $i_1 \succ i_2$, if $G_1 = G_2$ and all the remaining fields of i_1 dominates or are equal to the respective i_2 field, but one that must be strictly dominant. For the sake of brevity, we report only one of the formulas able to detects this anomaly:

$$s_1 \succ s_2 \wedge d_1 \succeq d_2 \wedge t_1 \succeq t_2 \wedge c_1^h \succeq c_2^h \wedge c_1^p \succeq c_2^p \wedge c_1^c \succeq c_2^c \wedge G_1 = G_2$$

Since the protection requirements of i_1 are 'greater' than the i_2 ones, the latter can be safely removed without altering the network semantic. However, an administrator can also choose to keep both the PIs, by following a security in depth approach.

The policy implementation i_1 is *affine* with the policy implementation i_2, and we will write $i_1 \not\perp i_2$, if:

$$(s_1 \not\perp s_2 \wedge d_1 \not\perp d_2 \wedge t_1 \not\perp t_2 \wedge G_1 = G_2) \wedge (i_1 \not\succeq i_2 \wedge i_2 \not\succeq i_1)$$

In short, we have an affinity when the two PIs are incomparable (i.e. neither dominant nor equivalent), the sources and destinations are on the same network node, and use technologies at different ISO/OSI layers. In order to solve these anomalies, i_1 and i_2 can be replaced with the least upper bound policy, that is, a new policy implementation $i_3 = (s_3, d_3, t_3, c_3^h, c_3^p, c_3^c, G_3)$ with the following fields:

- s_3 is the least upper bound of s_1 and s_2 in the network node tree, that is, $s_3 \succeq s_1$ and $s_3 \succeq s_2$ (see Fig. 2);
- d_3 is the least upper bound of d_1 and d_2 in the network node tree, that is, $d_3 \succeq d_1$ and $d_3 \succeq d_2$;
- t_3 is the least upper bound technology between t_1 and t_2, that is, $t_3 \succeq t_1$ and $t_3 \succeq t_2$;
- $c_3^h = c_1^h \vee v_2^h$, $c_3^p = c_1^p \vee v_2^p$ and $c_3^c = c_1^c \vee v_2^c$;
- $G_3 = G_1 = G_2$.

The policy implementation i_1 is an *alternative* to the policy implementation i_2 if all the fields of i_1 are equal to the fields of i_2, but $G_1 \perp G_2$, that is:.

$$s_1 = s_2 \wedge d_1 = d_2 \wedge t_1 = t_2 \wedge c_1^h = c_2^h \wedge c_1^p = c_2^p \wedge c_1^c = c_2^c \wedge G_1 \perp G_2$$

Two alternative PIs offers the same protection for the same end-points, but uses different communication paths, so that only one of them is really needed.

4 Implementation

In Section 3 we discussed our model and provided a set of FOL axioms which can be effectively used to detect the anomalies we identified. However, other equivalent representations are more usable and efficient, due to the availability of logical engines and other support tools.

We implemented the model presented in this paper as a set of Eclipse bundles in Java [5]. The plug-ins were developed during the PoSecCo FP7 project[6]. We tested our tool in two different scenarios:

- a small network with 18 nodes, 9 subnets, 5 gateways and 34 PIs;
- a medium-sized network with 37 nodes, 8 subnets, 7 gateways and 47 PIs.

We observed that, in the first case, the number of PI pair anomalies were 27: 9 inclusions, 4 alternatives and 14 affinities. While in the second case, are 52: 2 equivalences, 11 inclusions, 8 alternatives and 31 affinities. Taking into account that the number of conflicts varies mainly according to the network configuration chosen by the administrators. In both the cases the tool execution time was less than a second, proving that our model can be effectively used to analyse several kind of IT infrastructures.

In the following sections, we will discuss the implementation of our model using Horn clauses using the SWRL language and we will show a graph-based representation of the PI anomalies that can be helpful for the administrators to quickly identify such conflicts.

4.1 Anomaly conditions as horn clauses

Horn clauses are axioms defined as a disjunction of literals (clauses) with at most one positive literal, that is:

$$\neg C_1 \vee \neg C_2 \vee \cdots \vee \neg C_n \vee A$$

They can also be expressed in a more natural way as a set of positive conditions implying an assertion, that is:

$$C_1 \wedge C_2 \wedge \cdots \wedge C_n \Rightarrow A$$

These clauses can be effectively used to represent all the axioms used in our model. Horn clauses are frequently encountered in model theory because they exhibit a simple and natural rule-like form. Horn clauses can be then easily translated in many different logic programming languages, such as Prolog, or generic programming language such as C or Java.

For instance, the Horn clause form of the equivalence anomaly is:

$$\begin{aligned}&(s_1 = s_2) \wedge (d_1 = d_2) \wedge (t_1 = t_2) \wedge (c_1^h = c_2^h) \wedge \\ &(c_1^p = c_2^p) \wedge (c_1^c = c_2^c) \wedge (G_1 = G_2) \Rightarrow (i_1 = i_2)\end{aligned}$$

We implemented these axioms in an ontology, that naturally offers a hierarchical representation coupled with powerful inferential capabilities. We developed an ontology based on OWL 2 [8] and SWRL [9], which allows the specification of Horn-like rules that guarantee the computational soundness of the

[5] `http://security.polito.it/posecco/sdss/`
[6] `http://www.posecco.eu/`

reasoning. In the ontology, the policy implementations and the network fields are represented as individuals. These individuals are interconnected together through a series of object property assertions that provide the semantic of the relationships between the various network entities such as the belonging of an IP address to a network node. By using a set of ad-hoc SWRL rules we are able to automatically infer the anomalies as a set of property assertions. For example, the snippet in Listing 1.1 contains the SWRL rule for identifying equivalent PIs, by imposing, if needed, the property `equivalence` between two individuals representing a couple of PIs.

Listing 1.1: SWRL rule for detecting equivalent PIs.

```
hasSource(?i1,?s1), hasSource(?i2,?s2), hasDestination(?i1,?d1),
hasDestination(?i2,?d2), hasTechnology(?i1,?t1), hasTechnology(?i2,?t2),
hasHeaderIntegrity(?i1,?h1), hasHeaderIntegrity(?i2,?h2),
hasPayloadIntegrity(?i1,?p1), hasPayloadIntegrity(?i2,?p2),
hasConfidentiality(?i1,?c1), hasConfidentiality(?i2,?c2),
hasGw(?i1,?g1), hasGw(?i2,?g2), SameAs(?s1,?s2), SameAs(?d1,?d2),
SameAs(?t1,?t2), SameAs(?h1,?h2), SameAs (?p1,?p2), SameAs(?c1,?c2),
SameAs(?g1,?g2) -> isequivalence(?i1,?i2)
```

4.2 Graph-based representation of anomalies

In Section 2 we informally presented our hierarchical view of a network node (see also Fig. 2). By using such artifacts, we can depict a protected communication by the means of a multi-graph. The advantage of such graphical representation is that allows a network administrator to identify a series of anomalies in a more intuitive and natural way.

Our multi-graph representation includes a bush of network node trees which represent all the communication endpoints at network level and all the available gateways. Policy implementations are represented as paths that join together two vertices, that is, the source and the destination of the policy implementation. End-to-end communications are represented as a single edge path, on the other hand, tunnels require more edges to represent the communications to and from the gateways. To increase the graphical expressiveness, we also label each edge with the remaining policy implementation parameters, i.e. the technology and the security properties triple (with the trivial association f=false, and t=true).

With this graphical notation, PI pair anomalies are noticed by the presence of multiple paths between sources and destinations, while single PI anomalies can be noticed on the single edge.

We show here the graphical representation of two PI anomalies, namely an inclusion and an alternative conflicts taken from the simplified network scenario introduced in Section 2. Fig. 4 shows the anomaly between $i_{1,1}$ and $i_{1,3}$. The first

policy implementation $i_{1,1}$ is using a level 3 protection protocol without tunnels (e.g. IPsec via transport mode) while $i_{1,3}$ is another end-to-end connection but at application level (e.g. WS-Security). The multi-graph clearly shows that $i_{1,1}$ includes $i_{1,3}$, as there are two paths from the administrative assistant browser and the administrative web service and $i_{1,1}$ is at layer 3.

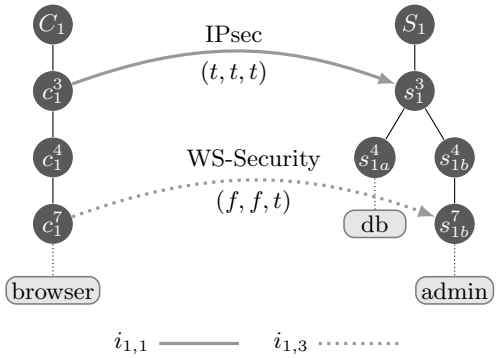

Fig. 4: Graphical representation of the anomaly $i_{1,1} \succ i_{1,3}$.

Fig. 5 depicts the alternative anomaly between $i_{c,s}$ and $i_{2,1}$. We recall that an alternative anomaly is a state where two policy implementations connects the same two entities, but use different network paths. In this case we have an end-to-end connection offered by $i_{2,1}$ and a site-to-site (VPN) communication given by $i_{c,s}$.

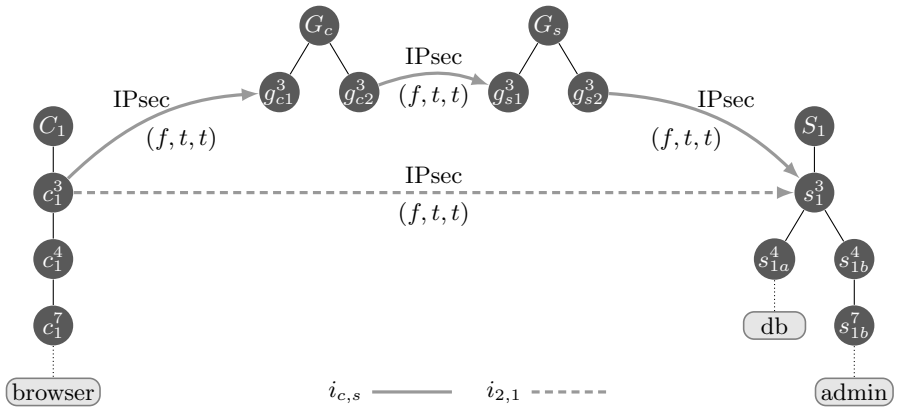

Fig. 5: Graphical representation of the anomaly $i_{c,s} \perp i_{2,1}$.

5 Related works

Conflict analysis, detection and resolution in policy-based systems and security controls is a hot topic and the current literature offers several notable works that we will briefly discuss in the following lines.

One of the most prominent works in this area is due to Al-Shaer et al., which address the analysis of filtering configurations [10], and take into account the effect of IPsec rules on a network protection [4], by proposing a number of ad-hoc algorithms and axioms. The same authors also describe a classification system for conflicts between filtering and communication protection policies in [3]. Furthermore, a common idea, initially introduced by Zao in [11], is to combine conditions that belong to to different IPSec fields. This was the basis used also in [12], where Fu et al. described a number of conflicts between IPSec tunnels, through a simulation process that reports any violation of the security requirements. Another interesting paper is [5], due to Li et al., where the authors classified the IPSec rules in two classes: access control lists (ACL) and encryption lists (EL). All these works treat only a single technology (IPsec), ignoring the possible interaction with other data communication protocols. Our model instead is able not only to detect single-technology anomalies, but also errors that can arise when deploying a multi-technology policy set.

Network configuration/policy anomaly detection is not only restricted to communication protection technologies. In literature a rich collection of papers about filtering policy analysis is also available. Although these works are not directly related to the approach presented in this paper, they can be very useful as a general background on network conflict analysis. In the following paragraphs we present a brief selection of several relevant works in this field.

Basile et al. describe a geometric representation, detection and resolution of filtering configurations, based on the intersection of hyper-rectangles [13]. Authors extended the work performed by Al-Shaer by introducing the anomalies between more than two rules and by showing how to transform a policy representation in another form that preserves its semantic. Similarly, Hu et al. in [14] suggested to split the five-tuple decision spaces of packet filtering rules into disjoint hyper-rectangles, where the conflicts are resolved using a combination of automatic strategies.

A thoroughly different approach for detecting conflicts between a set of filtering configurations is proposed by Hu et al., who introduced an ontology-based anomaly management framework [15], and by Bandara et al., who use logic reasoning, thus obtaining excellent performances [16].

Alfaro et al. presented a collection of algorithms to remove a series of anomalies between packet filter configurations and NIDS in distributed systems [17]. This techniques were more recently implemented in the MIRAGE tool[18].

With respect to policy conflict schemas for filtering rules, some interesting works also exists. Thanasegaran et al. show how to transform the configuration rules in bit vectors in order to have a very efficient analysis [19], while Ferraresi et al. extend al-Shaer's work and provide an automatic algorithm to resolve the anomalies [20].

6 Conclusions and future work

In this paper we proposed a novel taxonomy of anomalies for communication protection policies and an algebraic model which is able to detect such anomalies. The proposed model can be used to detect incompatibilities and redundancies between policy implementations that use security technologies working at different ISO/OSI layers.

Our model has been implemented using Horn clauses and ontological techniques and it can be also easily represented as a multi-graph, thus providing the administrators with a more intuitive way to identify the anomalies. Indeed, the model can be used as a tool to assist administrators when implementing communication protection policies. Moreover, since the model provides a number of hints about conflict resolution for each of the identified anomalies, our tool can be used to support automatic policy refinement.

For the future, we plan to extend the expressivity and capabilities of our model by taking into account also the adopted/supported cipher-suites and the actual paths walked by packets/messages in the network. The extended model will be able to take into account several new problems, such as channel overlapping misconfigurations (for VPN tunnels), potential information leakage and non-enforceable policy implementations.

Acknowledgement

The research described in this paper is part of the SECURED project, co-funded by the European Commission under the ICT theme of FP7 (grant agreement no. 611458).

References

1. A. Wool, "Trends in firewall configuration errors: Measuring the holes in swiss cheese," *IEEE Internet Computing*, vol. 14, no. 4, pp. 58–65, Jul. 2010.
2. Center for Strategic and International Studies, "Securing cyberspace for the 44th presidency," Tech. Rep., Dec. 2008.
3. H. Hamed and E. Al-Shaer, "Taxonomy of conflicts in network security policies," *IEEE Communications Magazine*, vol. 44, no. 3, pp. 134–141, Mar. 2006.
4. H. Hamed, E. Al-Shaer, and W. Marrero, "Modeling and verification of IPsec and vpn security policies," in *13th IEEE International Conference on Network Protocols*, ser. ICNP '05. IEEE Computer Society, Nov. 2005, pp. 259–278.
5. Z. Li, X. Cui, and L. Chen, "Analysis and classification of IPsec security policy conflicts," in *Japan-China Joint Workshop on Frontier of Computer Science and Technology*, ser. FCST '06. IEEE Computer Society, Nov. 2006, pp. 83– 88.
6. S. Kelly and S. Ramamoorthi, "Requirements for IPsec Remote Access Scenarios," RFC 3457, Jan. 2003.
7. A. Khakpour and A. X. Liu, "Quarnet: A tool for quantifying static network reachability," *IEEE/ACM Trans. Netw.*, vol. 21, no. 2, pp. 551 – 565, February 2009.

8. W. O. W. Group, "OWL 2 web ontology language document overview," Tech. Rep., Oct. 2009, http://www.w3.org/TR/2009/REC-owl2-overview-20091027/.
9. W3C, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," W3C Member Submission, World Wide Web Consortium, Tech. Rep., May 2004.
10. 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*, vol. 23, no. 10, pp. 2069–2084, Sep. 2006.
11. J. Zao, "Semantic model for IPsec policy interaction," Internet Draft, Tech. Rep., March 2000.
12. Z. Fu, S. F. Wu, H. Huang, K. Loh, F. Gong, I. Baldine, and C. Xu, "IPsec/VPN security policy: Correctness, conflict detection, and resolution," in *International Workshop on Policies for Distributed Systems and Networks*, ser. POLICY'01. Springer-Verlag, January 2001, pp. 39–56.
13. C. Basile, A. Cappadonia, and A. Lioy, "Network-level access control policy analysis and transformation," *IEEE/ACM Transactions Networking*, vol. 20, no. 4, pp. 985–998, Aug. 2012.
14. H. Hu, G.-J. Ahn, and K. Kulkarni, "Detecting and resolving firewall policy anomalies," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 3, pp. 318–331, May 2012.
15. ——, "Ontology-based policy anomaly management for autonomic computing," in *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*, ser. CollaborateCom. IEEE Computer Society, Oct. 2011, pp. 487–494.
16. A. K. Bandara, A. C. Kakas, E. C. Lupu, and A. Russo, "Using argumentation logic for firewall configuration management." in *Integrated Network Management-Workshops, 2009*, ser. IM'09. IEEE Computer Society, Jun 2009, pp. 180–187.
17. J. G. Alfaro, N. Boulahia-Cuppens, and F. Cuppens, "Complete analysis of configuration rules to guarantee reliable network security policies," *International Journal of Information Security*, vol. 7, no. 2, pp. 103–122, Mar. 2008.
18. J. Garcia-Alfaro, F. Cuppens, N. Cuppens-Boulahia, and S. Preda, "Mirage: A management tool for the analysis and deployment of network security policies," in *5th International Workshop on Data Privacy Management, and 3rd International Conference on Autonomous Spontaneous Security*, ser. DPM'10/SETOP'10. Springer-Verlag, set 2011, pp. 203–215.
19. S. Thanasegaran, Y. Yin, Y. Tateiwa, Y. Katayama, and N. Takahashi, "A topological approach to detect conflicts in firewall policies," in *IEEE International Symposium on Parallel & Distributed Processing*, ser. IPDPS'09. IEEE Computer Society, May 2009, pp. 1–7.
20. S. Ferraresi, S. Pesic, L. Trazza, and A. Baiocchi, "Automatic conflict analysis and resolution of traffic filtering policy for firewall and security gateway," in *International Chamber of Commerce*, ser. ICC 2007. IEEE Computer Society, Jun 2007, pp. 1304–1310.