Policy Enforcement Point Model

Yosra Ben Mustapha[✉], Hervé Debar, and Gregory Blanc

Telecom Sudparis, SAMOVAR UMR 5157, 9 rue Charles Fourier,
91011 Evry, France
{yosra.ben_mustapha,herve.debar,gregory.blanc}@telecom-sudparis.eu

Abstract. As information systems become more complex and dynamic, Policy Decision Points (PDPs) and Policy Enforcement Points (PEPs) follow the same trend. It becomes thus increasingly important to model the capabilities of these PDPs and PEPs, both in terms of coverage, dependencies and scope.

In this paper, we focus on Policy Enforcement Points to model the objects on which they may enforce security constraints. This model, called the PEP Responsibility Domain ($RD(PEP)$), is build based on the configuration of the PEP following a bottom-up approach. This model can then be applied to multiple use cases, three of them are shown as examples in this paper, including policy evaluation and intrusion detection assessment and alert correlation.

Keywords: Policy Enforcement Point · Approximation Accuracy · Alert correlation · Security policy

1 Introduction

Many policy enforcement mechanisms, herein referred to as Policy Enforcement Points (PEP), have been designed and developed in order to apply the access control decisions and protect the supervised network. Each policy enforcement mechanism is characterized by its capability. This capability encompasses both the kind of information it can collect to filter information (network adresses, emails, signatures) and on the kind of decision it can enforce (block or reject a request, send an alert, etc.). It also encompasses the position of the PEP in the information system, and its position in the processes that provide the service requested by the users. Thus, having a complete understanding of the coverage and capabilities of each enforcement mechanism is necessary to deploy it effectively, to evaluate its performance and to analyze its interactions with other PEPs.

We propose to model these Policy Enforcement Capabilities in order to have a good understanding of deployed Policy Enforcement capabilities and tackle several issues in security policy management and intrusion detection. This model is the PEP Responsibility Domain ($RD(PEP)$). The main objective of $RD(PEP)$ is to build a consistent view of the deployed policy enforcement capabilities that may contribute in defining the appropriate response decision. We first propose

J. Tian et al. (Eds.): SecureComm 2014, Part I, LNICST 152, pp. 278–286, 2015.
DOI: 10.1007/978-3-319-23829-6_20

a definition of a *Policy Enforcement Point Responsibility Domain* $RD(PEP)$. Second, we expose several approximation approaches of the RD(PEP). Third, we evaluate the differences between these approximations. Finally, we describe the application of the proposed PEP model on alert correlation.

2 Policy Enforcement Points

The term of Policy Enforcement Point (PEP) was introduced in [1] as an entity that performs access control by making decisions requests and *enforcing* authorization decisions by the Policy Decision Point (PDP). In [2], PEP is defined as the most security critical component, which protects the resources and enforces the PDP's decision. Generally, the PDP and the PEP are combined to control access and enforce the security policy. According to [3], PEPs are defined as modules which reside on the managed devices and are responsible for installing and enforcing the Security Policy.

In our approach, we define the PEP as a security entity that is capable to apply, on the triplet $\{Subject, Action, Object\}$, the enforcement decisions represented by $\{d_1, d_2, d_3, \ldots, d_p\}$ (p is the total number of all decisions that can be applied by the PEP class). In Eq. 1, we give an algebraic representation characterizing a PEP.

$$PEP : S \times A \times O \longrightarrow \{d_k\}_{k\in[1\ldots p]} \tag{1}$$

In general, the triplet {Subject, Action, Object} is represented by a set of appropriate attributes denoted by $\{Attr_i\}_{i\in[1\ldots n]}$. In the rest of our paper, we do not consider the decisions applied by the PEPs.

3 PEP Model

In this Section, we briefly define the basic notions used in our proposed approach.

3.1 Selector Definition

The security policy enforcement is usually based on a set of decision/enforcement criteria known as "Selectors". In general, a selector is a typed variable having a finite or infinite domain. We assume in our approach that all the selectors have a finite domain. This latter is denoted by $D(S)$. We denote by $\mid D(S) \mid$ the cardinality of D(S).

Selector Type. Each selector has a defined ***Selector Type*** denoted by $S.Type$. We define it by $S.Type = \{(Type(S), D(S))\}$. $Type(S)$ represents the type of the Selector S. It can be for example *integer, real, binary, string, timestamp, etc.*

Selector Domain Decomposition. Following the previous definition, $D(S)$ represents the range of all the possible values which can be affected to Selector S. $D(S)$ can be split into a finite number, l, of totally disjoint sub-domains denoted by $\delta(S)$. Those sub-domains are totally disjoint.

3.2 PEP Classes

We can distinguish between PEPs based on the communication stack layer: network-level (e.g. firewalls, routers, IDSes, IPSes), application-level (e.g. databases), or identity and access-level PEPs (directory access control). Hereafter, we introduce the notion of a *PEP class.*

Definition 1 ***PEP class:*** *A family of PEPs shares common functional characteristics and enforces the policy based on a common (sub)set of selectors.*

PEP Class Properties. Following the Definition 1, each class of PEP is characterized by an identical **core** set of *Selectors* denoted by $\{S_1, S_2, S_3, \ldots, S_n\}$.

4 Responsibility Domain of PEP Rules

This concept is related to the capability and ability for the PEP to enforce the Security Policy (SP). It is defined by the PEP's configuration and its rules. Let's m be the total number of configured rules. A rule r_i, for $i \in [1 \ldots m]$, has usually the following general form:

$$\begin{array}{ll} r_i : & C_i \rightarrow D_i \\ & \\ where & C_i : \ Conditions\ defined\ on\ Selectors \\ & C_i \ = \{r_i(S_j) = s_{ij}, \forall i \in [1 \ldots n]\} \\ and & D_i : \ set\ of\ decisions \end{array} \tag{2}$$

D_i are applied when C_i are satisfied. A rule can apply several decisions such as denying and logging. Every rule r_i, $\forall i \in [1 \ldots m]$, defined in the PEP configuration has an explicitly defined Responsibility Domain.

Definition 2 ***Responsibility Domain of a rule:*** *It is derived from the set of C_i configured for each* Selector *of the PEP. It includes all of the packets, requests, etc., on which the rule's decision(s) may be applied and enforced. We denote it by $RD(r_i)$ as is written as follow:*

$$RD(r_i) = \ < s_{ij} >_{j \in [1 \ldots n]}, 1 \leq i \leq m \tag{3}$$

Since $s_{ij} \subseteq D(S_j)$, one rule may include different selectors combination. We define hereafter the $RD(r_i)$ coverage.

Consequence 1 $RD(r_j)$ ***Coverage:*** $RD(r_i)$ *Coverage is the number of all the selectors combination defined in the rule r. It is expressed in Eq. 4.*

$$\mid RD(r_i) \mid = \prod_{j \in [1 \ldots n]} \mid s_{ij} \mid \tag{4}$$

Example. Consider the following rule of a Network-level Firewall:

$$\begin{array}{l} r : src\_ip = 140.192.37. * \wedge src\_port = * \wedge \\ \quad dst\_ip = 161.120.33.40 \wedge dst\_ip = 80 \wedge protocol = tcp \\ \quad \rightarrow deny \end{array} \tag{5}$$

Its corresponding Responsibility Domain is:

$$RD(r) =< 140.192.37.*, *, 161.120.33.40, 80, tcp > \tag{6}$$

and the coverage of $RD(r)$ is:

$$\begin{aligned} \mid RD(r) \mid &=\mid 140.192.37.* \mid, \mid * \mid, \mid 161.120.33.40 \mid, \mid 80 \mid, \mid tcp \mid> \\ &= (2^8 - 1) \times (2^{16} - 1) \times 1 \times 1 \times 1 \end{aligned} \tag{7}$$

4.1 Characterization of Relations Between $RD(r_j)$

In [4], authors define five relations that may exist between rules (Fig. 1). They demonstrate that these relations are unique and that can be applied to define the different conflicts and anomalies that may figure between rules. We adopt these relationships and define them between Responsibility Domain of rules. Overlaps between rules result in overlaps between their Responsibility Domains. Hereafter, we detail the relationships that may exist between the Responsibility Domains of rules.

- $RD(r_1)$ and $RD(r_2)$ are ***Completely Disjoint*** (CD) and we write $CD(r_1, r_2)$, iff

$$\begin{aligned} \forall j \in [1 \ldots n],\ r_1(S_j) \ntrianglerighteq r_2(S_j) \\ where \trianglerighteq \in \{\subset, \supset, =\} \end{aligned} \tag{8}$$

- $RD(r_1)$ and $RD(r_2)$ are ***Exactly Matched*** (EM) and we write $EM(RD(r_1), RD(r_2))$, iff

$$\forall j \in [1 \ldots n],\ r_1(S_j) = r_2(S_j) \tag{9}$$

- $RD(r_1)$ and $RD(r_2)$ are ***Inclusively Matched*** (IM) and we write $IM(RD(r_1), RD(r_2))$, iff

$$\begin{aligned} &\forall j \in [1 \ldots n], &&r_1(S_j) \subseteq r_2(S_j) \\ &and\ \exists j'\ such\ that\ &&r_1(S_{j'}) \neq r_2(S_{j'}) \end{aligned} \tag{10}$$

- $RD(r_1)$ and $RD(r_2)$ are ***Partially Matched*** (PM) and we write $PM(RD(r_1), RD(r_2))$, iff

$$\begin{aligned} &\exists j',\ j'' \in [1 \ldots n],\ &&r_1(S_{j'}) \trianglerighteq r_2(S_{j'}) \\ & &&r_1(S_{j''}) \ntrianglerighteq r_2(S_{j''}) \\ &where &&\trianglerighteq \in \{\subset, \supset, =\} \end{aligned} \tag{11}$$

- $RD(r_1)$ and $RD(r_2)$ are ***Correlated*** (C) and we write $C(RD(r_1), RD(r_2))$, iff

$$\begin{aligned} &\forall j \in [1 \ldots n], &&r_1(S_j) \trianglerighteq r_2(S_j) \\ &and\ \exists\ j',\ j''\ \in [1 \ldots n]\ &&such\ that\ r_1(S_{j'}) \subset r_2(S_{j'}) \\ & &&and\ r_1(S_{j''}) \supset r_2(S_{j''}) \\ &where &&\trianglerighteq \in \{\subset, \supset, =\} \end{aligned} \tag{12}$$

Contrary to [4], these relationships are used in order to set approximation inferences that will be detailed in Sect. 5.

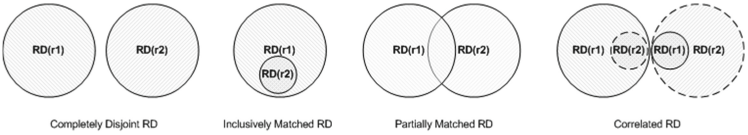

Fig. 1. Relations between Responsibility Domains of two rules.

5 Responsibility Domain of PEP

5.1 Axioms

Before detailing our proposed approach, it is important to define the assumptions that constitute a *sine qua non* condition to develop our approach.

- **Axiom 1.** All considered PEPs have a finite set of rules. In practice, security administrators configure on each PEP a finite set of rules which apply the Security Policy Guidelines.
- **Axiom 2.** We ignore the default rule of the PEP. Usually, since the default rule includes the entire selectors domains, it does not inform us about the configuration specification of the PEP.
- **Axiom 3.** The definition of the Responsibility Domain should only consider the intrinsic characterization and deployment of the PEP.

5.2 Definitions

Definition 3 ***Responsibility Domain of PEP:*** *Each PEP, once deployed in the network, has a finite range of applicability which we call "Responsibility Domain". The Responsibility Domain of the PEP informs us about the enforcement capability of the PEP across the network. We denote it by $RD(PEP)$. This domain is an abstraction over the PEP implementation and configuration and its intrinsic enforcement capabilities.*

Definition 4. *The Responsibility Domain is a bounded multi dimensional domain and its dimension is $Dim(PEP)$.*

The $RD(PEP)$ is a bounded domain. We respectively denote the upper bound and the lower bound by $RD_{sup}(PEP)$ and $RD_{inf}(PEP)$. $RD_{sup}(PEP)$ considers environmental constraints on the deployed PEP. The identification of this bound requires external knowledge related to the topological visibility of the deployed PEP. $RD_{inf}(PEP)$ is the union of the entire set of Responsibility Domains of configured rules in the PEP's instantiation.

As policy enforcement is, in most cases, distributed along the different PEPs, it is important to model their enforcement capability, $RD(PEP)$, in order to support the administrator in selecting the most appropriate ones. Thus, the definition and identification of an appropriate approximation of the $RD(PEP)$ must be well defined. Hereafter, we first give an identification of $RD_{inf}(PEP)$ and then detail several approximations of the $RD(PEP)$.

5.3 Definition of $RD_{inf}(PEP)$

We refer to the configuration matrix $Conf_{selectors}(PEP)$ defined in Eq. 13. It not only represents the configuration of the PEP but also identify the $RD_{inf}(PEP)$. $RD_{inf}(PEP)$ is the union of the entire set of Responsibility Domains of configured rules in the PEP's instantiation.

$$Conf_{selectors}(PEP) = \begin{array}{c} \\ r_1 \\ r_2 \\ \vdots \\ r_m \end{array} \begin{array}{c} \begin{array}{cccc} s_1 & s_2 & \ldots & s_n \end{array} \\ \begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1n} \\ s_{21} & s_{22} & \cdots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & \cdots & s_{mn} \end{pmatrix} \end{array} \tag{13}$$

The definition of $RD_{inf}(PEP)$ takes into account the entire set of the different combinations between selectors defined in configured rules while ignoring the default rule.

$$\begin{aligned} RD_{inf}(PEP) &= \bigcup_{i \in [1 \ldots m]} RD(r_i) \\ &= \bigcup_{i \in [1 \ldots m]} < s_{ij} >_{j \in [1 \ldots n]} \end{aligned} \tag{14}$$

Objective of $RD(PEP)$ approximations and methodology

Our objective at this stage is to analyze different possibilities of a comprehensive and appropriate approximation of $RD(PEP)$ without losing specificities of the deployed PEP. The $RD_{inf}(PEP)$ is considered as the unique starting point for all the approximations. Based on $RD_{inf}(PEP)$, we define inferences and data mining operations to build different versions of approximated Responsibility Domains denoted as $RD_{apprx}(PEP)$. These operations consider the relations between rules and characterizations of selectors, their combination properties. We detail them in the next two paragraphs.

The different approximations that we propose can be split in two major categories:

- Rule-based *(rb)* approximations, Eq. 15: It is based on rules which are represented by the rows of $Conf_{selectors}(PEP)$ matrix.

$$rb() : \begin{array}{c} \mathcal{U} \longrightarrow \mathcal{U} \\ RD_{inf}(PEP) \longmapsto RD_{rb\_apprx_1(PEP)} \end{array} \tag{15}$$

  The first rule-based approximation $RD_{rb\_apprx_1}(PEP)$ is the result of the function $rb(RD_{inf})$.

- Selector-based *(sb)* approximations, Eq. 16: It is based on values affected to selectors across columns of $Conf_{selectors}(PEP)$ matrix.

$$sb() : \begin{array}{c} \mathcal{U} \longrightarrow \mathcal{U} \\ RD_{inf}(PEP) \longmapsto RD_{sb\_apprx_1}(PEP) \end{array} \tag{16}$$

  The first selector-based approximation $RD_{sb\_apprx_1}(PEP)$ is the result of the function $sb(RD_{inf})$.

Due to space limitation, $rb()$ and $sb()$ functions will not be detailed.

$$gen(): \begin{array}{c} \mathcal{U} \longrightarrow \mathcal{U} \\ < s_j, j \in [1 \ldots n] > \longmapsto < \delta_{k_j}, j \in [1 \ldots n] > \end{array} \tag{17}$$

This function, gen(), refers to a generalization process which considers the Selector Domain Partition defined in Sect. 3.1. For each selector instantiation, s_j, gen() identifies the corresponding sub-domain including the partition of s_j. The resulting vector will be a tuple of these generalized partitions.

6 Analysis of *RD*(*PEP*) Approximations and Interpretations

6.1 *RD*(*PEP*) Approximations Properties

As explained above, all the approximations closely depends on the configuration of the deployed *PEP*. Therefore, several relations would exist between these approximations:

- **Totally Inclusive Approximations**: The application of $gen()$ function results in a generalization of considered selectors values.

$$\begin{array}{c} RD_{rb\_apprx_1}(PEP) \subseteq RD_{rb\_apprx_2}(PEP) \\ \Rightarrow \mid RD_{rb\_apprx_1}(PEP) \mid \leqslant \mid RD_{rb\_apprx_2}(PEP) \mid \end{array} \tag{18}$$

- **Partially Joint Approximations**: Both of $RD_{rb\_apprx_2}(PEP)$ and $RD_{sb\_apprx_1}$ may have a common set of vectors which is at least the $RD_{rb\_apprx_1}$.

6.2 Qualitative Analysis: Approximation Accuracy Metric

The evaluation results shown in this paragraph are based on the approximations of $RD(Firewall)$ based on $RD_{inf}(Firewall)$ of the following running example shown in Fig. 2. It represents a medium size network with two zones (D1 and D2) connected to the Internet and protected by a border *Firewall* which is an instantiation of netFW class.

Identification of $RD_{sup}(Firewall)$: The RD_{sup} of a deployed PEP includes the set of all possible vectors characterizing the flow that may pass through the PEP. We denote by $D_{sup}(S)$ the *real* domain of a Selector S. It is identified by considering the topological information about the network.

$$\begin{array}{lll} D_{sup}(src\_ip) & = & \{140.192.37.*,\ 161.120.33.*,\ *.*.*.*\} \\ D_{sup}(dst\_ip) & = & \{140.192.37.*,\ 161.120.33.*,\ *.*.*.*\} \\ D_{sup}(p) & = & \{tcp,\ udp\} \end{array} \tag{19}$$

$$\begin{aligned} RD_{sup}(Firewall) = \{ & D_{sup}(src\_ip) \\ & \times D_{sup}(dst\_ip) \times D_{sup}(p), \\ & such\ that: \\ & \{D_{sup}(src\_ip), D_{sup}(dst\_ip), \\ & D_{sup}(p)\} \text{ are combinable} \end{aligned} \tag{20}$$

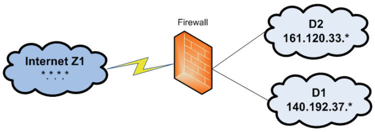

Fig. 2. Running Example: a medium size network with two zones.

Approximation Accuracy has been introduced in several mathematical theories such as approximation theory, rough set, fuzzy set, etc. In our approach, we propose to apply this metric in order to evaluate how accurate the approximations are regarding the *real* Responsibility Domain of the PEP. We adapt the Approximation Accuracy expression defined in *Rough Set Theory* [5]. In [5], the author define the Accuracy Approximation as a measure to express the *quality* of the approximation.

In Eq. 21, we define the Approximation Accuracy of $RD_{apprx}(PEP)$ which we denote as $\Lambda(RD_{apprx}(PEP))$ as:

$$\Lambda(RD_{apprx}(PEP)) = \frac{|RD_{apprx}(PEP)|}{|RD_{sup}(PEP)|} \tag{21}$$

Obviously, $0 < \Lambda(RD_{inf}(PEP)) \leq \Lambda(RD_{apprx}(PEP)) \leq 1$ for any $RD_{apprx}(PEP)$. Following the definition of the approximation of the Responsibility Domain, the more $\Lambda(RD_{apprx}(PEP))$ is closer to 1, the more accurate the approximation is.

In Table 1, we evaluate this metric for the different approximations of the running example. Based on results shown in Table 1, we notice that the Approximation Accuracy of approximations $RD_{sb\_apprx_1}$ and $RD_{sb\_apprx_2}$ is 10^8 times bigger than the $\Lambda(RD_{inf}(PEP))$. In this case, selector-based approximations are more appropriate than rule-based approximations.

Table 1. Evaluation of Approximation Accuracy Metric of the running example

	RD_{inf}	$RD_{rb\_apprx_1}$	$RD_{rb\_apprx_2}$	$RD_{sb\_apprx_1}$	$RD_{sb\_apprx_2}$
$\Lambda(RD_{apprx}(PEP))$	$4 * 10^{-11}$	$2,5 * 10^{-8}$	$1,3 * 10^{-3}$	$3,9 * 10^{-3}$	$3,9 * 10^{-3}$

7 Application on Alert Correlation

The proposed PEP model can be used in several security applications. Hereafter, we detail one of the novel applications of such PEP model.

The Responsibility Domain of deployed PEPs is considered as a correlation feature. Alerts are correlated if they share a common (set of) PEP(s) capable of applying a countermeasure on the corresponding flow of the alert. Hereafter, we define our proposed Enforcement-based Alert Correlation.

Definition 5 ***Enforcement-based Alert Correlation:*** *Given a set of alerts and a set of Responsibility Domains of the deployed PEPs, the Enforcement-based Alert Correlation groups alerts while considering the Responsibility Domains of PEPs, as the correlation feature.*

In Eq. 22, we write the basic correlation inference used in our Enforcement-based Alert Correlation approach for two different alerts A_1 and A_2.

$$\begin{aligned} A_1 \in RD(PEP_1) \wedge A_2 \in RD(PEP_1) \\ \Longrightarrow A^{ec} = \langle (A_1,\ A_2), (PEP_1) \rangle \end{aligned} \tag{22}$$

A^{ec} represents the Enforcement-based Correlated Alert. It is composed of two components. The first component includes the set of correlated alerts. The second component includes the (set of) PEP(s) that is (are) capable to process the correlated alerts.

A^{ec} is intended to group one or more previously-sent alerts together, to say "these alerts can be processed by the common PEP(s)". This application shows how our model is capable to enhance the response decision process.

8 Conclusions

We introduce a novel concept to model Policy Enforcement Point by their Responsibility Domain, $RD(PEP)$. We first characterize the PEP by the set of selectors. Then, we define the Responsibility Domain of a configured rule $RD(r)$. We analyze the relationships that may exist between these domains and define a set of approximation inferences. Based on the different properties that exist between $RD(r)$ and the characterization of selectors, we give different approximations of the $RD(PEP)$. The advantage of our methodology to approximate $RD(PEP)$ is the performance in a 'blind manner'. Also, the consideration of the PEP configuration makes the approximations more useful for response decision. Our future work is mainly oriented toward studying the different properties that may exist between these approximations of different deployed PEPs. The main objective would be the application of this model in a distributed response decision and alert correlation.

References

1. eXtensible Access Control Markup Language (XACML) (2003). https://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf
2. Zaborovsky, V., Mulukha, V., Silinenko, E.: Access Control Model and Algebra of Firewall Rules
3. Boutaba, R., Polyrakis, A.: Towards extensible policy enforcement points. In: Sloman, M., Lobo, J., Lupu, E.C. (eds.) POLICY 2001. LNCS, vol. 1995, pp. 247–262. Springer, Heidelberg (2001). http://dl.acm.org/citation.cfm?id=646962.712111
4. Al-shaer, E.S., Hamed, H.H.: Discovery of policy anomalies in distributed firewalls. In: IEEE INFOCOM 2004, pp. 2605–2616 (2004)
5. Pawlak, Z.: Rough Sets. Int. J. Inf. Comput. Sci. **11**, 341–356 (1982)