Ontology-based Security Policy Translation

Cataldo Basile, Antonio Lioy, Salvatore Scozzi, and Marco Vallini

Politecnico di Torino
Dip. di Automatica e Informatica
Corso Duca degli Abruzzi, 24, 10126 Torino (Italy)
{cataldo.basile, antonio.lioy, marco.vallini}@polito.it, salvatore.scozzi@gmail.com

Abstract: The security configuration of large networked ICT systems is a difficult and error-prone task. Quite often attacks are enabled by mis-configurations generated by human errors. Policy-based network management has been proposed to cope with this problem: goals are expressed as high-level rules that are then translated into low-level configurations for network devices. While the concept is clear, there is a lack of tools supporting this strategy. We propose an ontology-based policy translation approach that mimics the behaviour of expert administrators, without their mistakes. Normally administrators are given the high-level security goals and then, through their knowledge of network topology and security best practice, derive the device configurations. In a similar way, we use an ontology to represent the domain knowledge and then perform reasoning (based on best practice rules) to create the configurations for network-level security controls (e.g., firewall and secure channels). If some information is missing from the ontology, the administrator is guided to provide the missing data. The configurations generated by our approach are represented in a vendor-independent format and therefore can be used with several kinds of devices.

Keywords: security policy, policy-based network management, automatic policy translation, ontology-based translation, security ontology**.**

1. Introduction

Currently one major weakness in today security landscape is still an old one: the human factor. Already back in the year 2000, the “Roadmap for defeating Denial of Service attacks”[1] highlighted the general scarce technical talent and decreasing competence of system administrators. Unfortunately, this scenario has not significantly improved, as confirmed by recent reports [1]. It is therefore important to support security managers with automatic tools to minimize human errors.

Presently, policy-based network management (PBNM) [2] [3] seems the best approach to cope with system administration because it separates the goals (i.e., the policy) from the mechanisms to achieve them (i.e., the security controls).

Policies are expressed as high level security statements, derived from business goals or best-practice rules. However actual security controls are mostly placed at network level. Firewall and virtual private network (VPN) are ubiquitous in security architectures because they have excellent performance (compared to application-level controls), apparently are easy to configure, and do not require changes to business applications. Evidence says that as the network grows bigger, the configuration of network-level security controls becomes exponentially complex. We have therefore a mismatch: policies are expressed at high levels while controls are at network level.

Bridging this gap is the task of policy translation that consists in transforming a policy from the level of abstraction A to the (lower) level B. This procedure is repeated until the policy is expressed in a way that can be used to configure a specific control (e.g., IP-level filters for a firewall). Quite often translation from high-level policies to actual controls is manually performed by security administrators but this is still time-consuming and error prone: if policy translation is not automated, PBNM is simply a problem shift, not a solution. Our approach, called Ontology-Based Policy Translator (OPoT), handles the problem of policy translation using ontology-based reasoning to refine policies into configuration for the actual controls.

In particular, we aim at creating system able to mimic skilled administrators in collecting all the needed information and applying best practice for security configuration. Ontologies are very effective in capturing, defining, sharing, and reusing the knowledge about a specific domain. In particular, they have proved effective for representing relationships between entities and for verifying the correctness of knowledge [4]. Therefore we use ontology to represent the domain of interest - computer networks - from the topological and functional point of view, and the environmental data necessary to correctly configure them.

OPoT exploits ontology-based reasoning to understand the information necessary for policy translation and to enrich the knowledge base in order to fill the gap between abstraction levels. In other words, automatic reasoning is used to enlarge the domain of knowledge of the ontology without the need to involve human administrators. Nevertheless, the reasoning alone is unable to fully translate a policy because some information cannot be guessed or derived (e.g., the users' roles). To this purpose, OPoT guides the administrator in acquiring the data necessary to complete the configurations and forces them to collect the right information. OPoT also reports about possible anomalies when it discovers that best practice is not followed.

The advantages of this approach are manifold. First, it addresses policy translation in a way that reduces the impact of administrators' mistakes in the configuration workflow, we just assume that the information they provide (e.g., organization charts, tasks) is correct. Second, it strongly decreases the effort and the time to have a set of correct configurations (from days to minutes). Last but not least, it quickly adapts to changes in the methodologies.

[1] http://www.sans.org/dosstep/roadmap.php

Received January 17, 2010

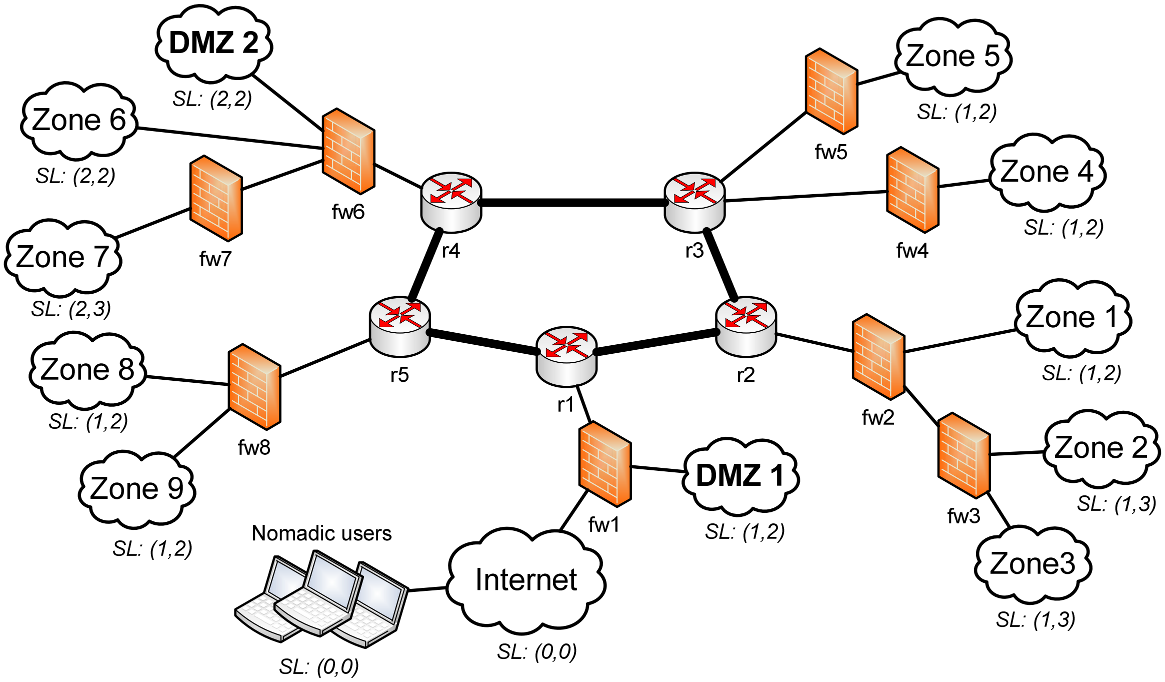


Figure 1. The reference network used to validate our approach.

The proposed approach can still be improved in several aspects, but our initial experiments demonstrate its power as well as its extensibility to new policies and reasonings.

The paper is organized as follows: Section 2 briefly presents the most relevant works about usage of ontology for policy management; Section 3 illustrates the case study we will use thorough the whole paper for presenting and validating our model; Section 4 presents in detail our approach to ontology-based policy refinement; Section 5 shows an entire usage of our tool to the reference example; Section 6 presents the tool's implementation and testing results; finally, Section 7 draws conclusions and gives indications for future works.

2. Background and related work

The adoption of security ontology is continuously increasing for network security and risk management fields. Strassner suggests that the traditional information and data models are not capable to describe detailed semantics required to reason about behaviour. His work modifies the existing DEN-ng policy model to support and generate ontology for governing behaviour of network devices and services [5].

Tsoumas et al. develop security ontology to perform network security management thus modelling assets (e.g., data, network, and services), countermeasures (e.g., firewall, antivirus) and the related risk assessment [6]. They also describe a framework for security knowledge acquisition and management. This is composed by the following steps: populating the security ontology, gathering infrastructural data (e.g., network topology, services), collecting the security requirements, defining the security controls (derived from security requirements), and finally enforcing policies and monitoring system policy changes.

Fenz et al. define a knowledge model (defining a security ontology) to support risk management domain, incorporating security concepts like threats, vulnerabilities, assets, controls, and related implementation [7]. Another work [8], proposes an ontology-based approach to model risk and dependability taxonomies. The related framework contains a tool for simulating threats against the modelled ontology.

KAoS represents another approach to policy-based management. It uses a description logic and ontology for policy representation, conflict analysis and translation [9] [10].

3. Case Study

We present here the reference network that will serve as a validation of our approach and to present main results and implementation. This example represents a medium size single domain of administration network. In this network we will configure two categories of security protections: filtering and channel protection policies. Enforcing these kinds of policies is theoretically easy; nevertheless they are constant sources of problems and mistakes, especially when the size on the network grows and for non experienced administrators [11].

Figure 1 shows the network environment of a public organization. The network core is composed by 5 routers (r1, r2, r3, r4, r5) and is connected to the Internet by the firewall fw1. The entire system contains 100 nodes divided into the following categories: workstation, server, router, firewall and switch. A server is a computer that provides a service through a Service Access Point (SAP). A workstation instead is a computer that consumes a service provided by a server.

The network is organized into different IP sub-networks, according to different organization's departments and it includes also two DMZ (De-Militarized Zone). The network is also divided into 9 filtering domains, called zones [12]. Each zone is delimited by network elements with filtering capability (firewalls in this case study).

Zones and DMZs are associated to a Security Level (SL). More in detail, the SL is the pair expressing the number of DMZs encountered and the number of firewall(s) that are traversed from Internet (at SL=(0,0)) to end-nodes (e.g., servers and workstations).

Figure 2 represents the detailed descriptions of DMZ 1, DMZ 2 and Zone 7 which contain the set of services to support organization tasks and activities. DMZ 1 is connected through the fw1 to the external world and offers services available from Internet. In particular web-svr1, ap-svr1 and db-svr1 compose the web application that interacts with the external users; mail-svr1 is the system hosting the organization e-mail domain. Instead, the DMZ 2 contains services of organization information system (composed by ap-svr2 and web-svr2) which are only available for internal users. The Zone 7 contains a database server (db-svr2) and two file servers (storage-svr1, storage-svr2). The db-svr2 hosts the confidential data that regards the organization information system and interacts with ap-svr2 (DMZ 2). Finally the storage-svr1 and storage-svr2 offer file sharing and backup services to organization users.

The other zones contain only workstations and laptops, in particular, Zone 4 contains the guest network and Zone 2 and Zone 3 hosts IT administration department. The nomadic users are organization employees and managers who can access to information system services from Internet.

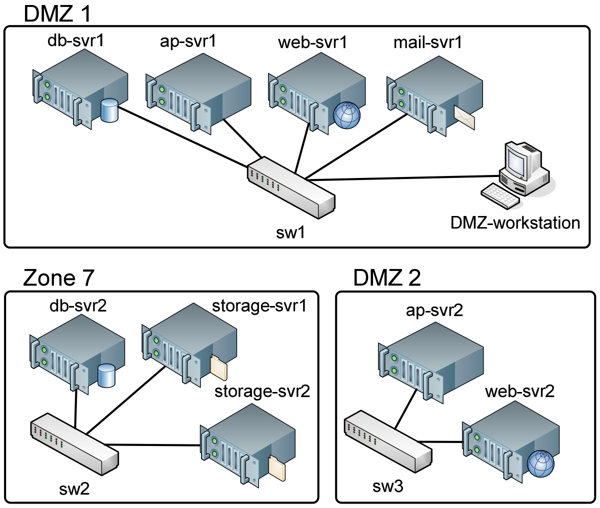


Figure 2. Details of the two DMZs and the Zone 7.

4. Our approach

The objective of our ontological framework is to derive configurations for security controls - based on ACL (Access Control List) and secure channel mechanisms - from a fixed set of business-oriented policies using automatic methods and interactions with the security administrator.

The security administrator is the person in charge of configuring network security. He is the intended user of our tool. We assume that he can access all the information the tool will ask him to generate configurations.

The ACL controls are derived both for devices (e.g., router, managed switch, firewall) and end-nodes. On the end-nodes we can distinguish two classes of controls: OS-level (e.g., local firewall) and application-level (e.g., Apache ACL configuration). The secure channel controls could be classified as: end-to-end, gateway-to-gateway and mixed (end-node-to-gateway, as in the case of VPN external access). In particular, we aim at configuring SSL/TLS end-to-end channels and IPsec channels end-to-end, gateway-to-gateway and end-node-to-gateway modes.

To improve the flexibility and adaptability of the tool, the output of the ontology-based policy translation process is represented in a vendor-independent format and stored as properties of the ontology to be translated in the correct format in a successive step.

The OPoT architecture is presented in Figure 3b. It is possible to highlight four different parts:

- the *input area* manages the OPoT's access to the configuration files (where the system description and the list of available policies are stored), parses them and populates the ontology with the initial values;
- the *user area* contains the graphical user interface that permits interaction with the administrator;
- the *knowledge management area* includes all the components that actually perform the policy translation;
- the *output area* produces the rule sets for the devices that must actually enforce the chosen policy in given network according to the chosen formats.

In the following sections all these parts are presented more in detail.

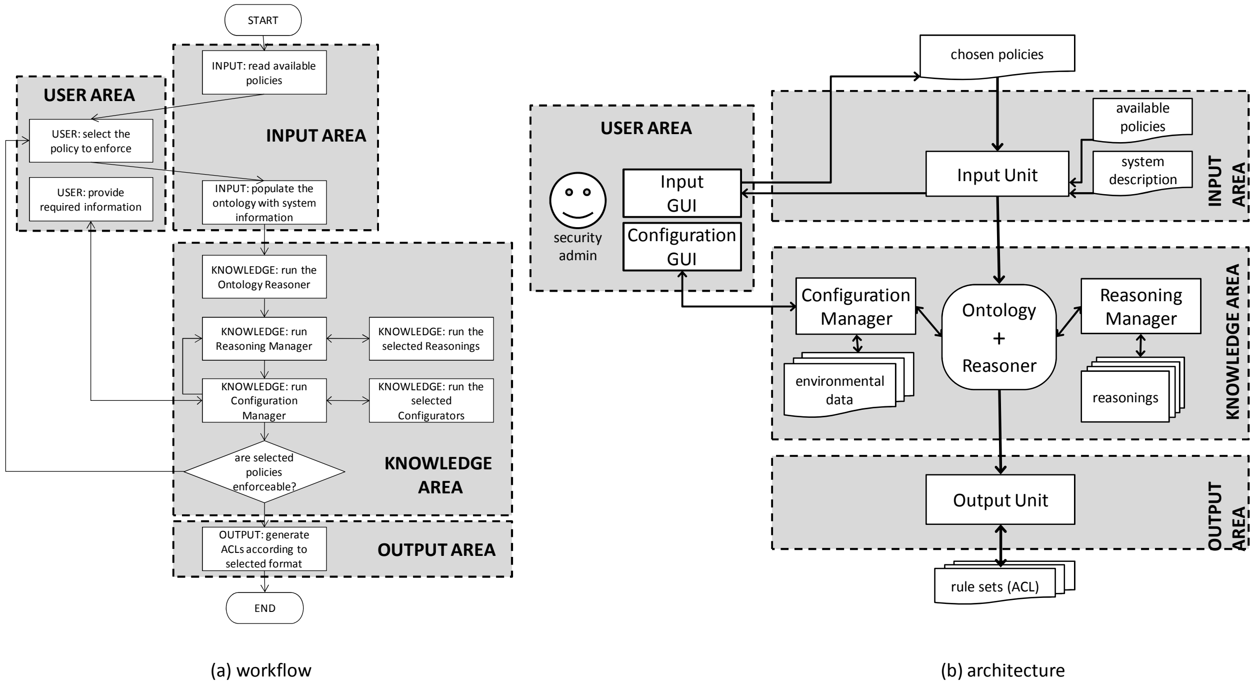
4.1 Input area

The Input Unit is in charge for managing these inputs. Its modular architecture permits to easily switch between different inputs formats. The inputs of the ontology-based policy refinement process are the set of the available policies and the system description.

Every available policy is a statement describing a business-level security target. Example of policies are "every user must access only the services he is authorized to use", or "business services must be connected securely to their components located in other parts of the network at higher security".

We analyzed different sources to decide the sample policies to support. In fact, in our intentions, the sample policies have to be a valid example of company's goals that administrators are typically asked to enforce by configuring the security controls available in their networking environments.

We combined our experience and interviews with the security administrators in our institution (representative of an open environment with several unrelated actors, multiple levels of security and untrusted users) and partners (representative of private networks handling highly sensitive data and hosting some public services). Additionally, we analyzed different policy models, such as Role Based Access Control (RBAC) [13] and Mandatory Access Control [14]. Finally, we examined publicly available policy templates, in particular the SANS Security Policy Project [15]. In this proof of concept, the configurations are obtained as


START
INPUT: read available policies
USER AREA
USER: select the policy to enforce
USER: provide required information
INPUT AREA
INPUT: populate the ontology with system information
KNOWLEDGE: run the Ontology Reasoner
KNOWLEDGE: run Reasoning Manager
KNOWLEDGE: run the selected Reasonings
KNOWLEDGE: run Configuration Manager
KNOWLEDGE: run the selected Configurators
are selected policies enforceable?
KNOWLEDGE AREA
OUTPUT: generate ACLs according to selected format
OUTPUT AREA
END
chosen policies
USER AREA
security admin
Input GUI
Configuration GUI
available policies
Input Unit
system description
INPUT AREA
Configuration Manager
Ontology + Reasoner
Reasoning Manager
environmental data
reasonings
KNOWLEDGE AREA
Output Unit
OUTPUT AREA
rule sets (ACL)

(a) workflow

(b) architecture

Figure 3. The workflow and architecture of the ontology-based translation process.

translation of a set of eight policies. See Section 4.2 for the full list of sample policies and their descriptions

Together with policies, the Input Unit populates the initial security ontology (see Section 4.4) available at the knowledge management area with the provided system description (i.e., topological and functional arrangement of the network).

4.2 Supported policies

This section presents and discusses the selected policies together with a short description about how administrators usually enforce them and how OPoT approaches their automatic enforcement.

(P1) organization users must be allowed access to the (organization) information system resources.

To enforce this policy, an administrator should identify which users are authorized to consume a particular service. Generally an organization contains several type of users divided into groups with different roles (e.g., employee, manager, CEO, etc.). This information is typically defined using a role-based approach, where each role can perform a set of operations on a service. Hence the administrator, considering user's role, should derive: 1) the user's workstation and the related IP address; 2) IP address, protocol and port of the service. In the next step he analyses the network topology and selects the filtering devices to allow authorized traffic. Finally he derives the specific configuration for each selected device. The automatic tool performs the same operations organized in different processes to achieve the same result. In particular a subset of these operations (network services analysis, public services analysis), are performed automatically, warning the administrator in case of uncertainty. The users and associated roles, if not explicitly described in an "official" document, are requested to the administrator.

(P2) organization external users (nomadic users) have to access securely to the (organization) information system resources.

The enforcement of this policy requires steps similar to the policy P1 but it involves different entities. However, due to the fact that the external users are in a public network, the administrator should adopt a channel protection mechanism. This is necessary to guarantee the integrity and the confidentiality of the traffic among networks with different security levels. Consequently, the filtering devices are configured to permit the protected channel. A typical approach to cope this scenario is to use a VPN in remote access mode, for example using the IPsec technology. In that case, the administrator picks the IPsec-capable device to configure the VPN among the internal network and external user's client. The automatic tool, as for P1, performs automatically the same operations. In particular, the tool analyses the network device capabilities and functionalities to identify which devices support the available channel protection and which are configurable for this purpose.

(P3) organization users have to access the Internet resource.

The enforcement of this policy permits the configuration of the Internet access to the organization users. It requires, as in P1, the identification of user's IP address to configure the related filtering devices. In addition, it is possible authorize only a subset of public services (e.g., web navigation using HTTP(s) protocol). In that case the tool asks to administrator which Internet services should be available for internal users. For example, an administrator could specify that P2P traffic (using UDP protocol) should be blocked.

(P4) administrators must be allowed access to the services they manage.

The enforcement of this policy requires that the administrator, who is responsible to manage network

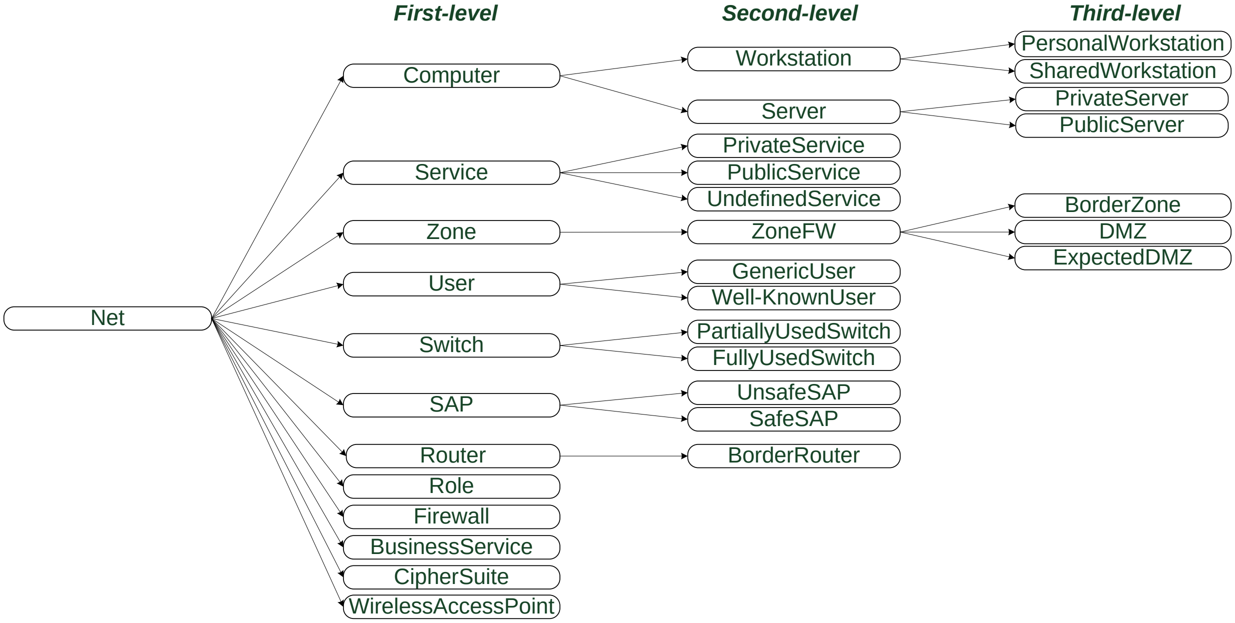


Figure 4. An extract of the security ontology.

services and devices, configures the access controls for the services under his management domain. The information about technical service (e.g., SSH) and devices to manage are generally derived from documents or network analysis. The implementation of P4 requires the configuration of the filtering devices involved in a communication between the administrator’s workstation and service/host/device to manage. The automatic tool analyses network description and shows to the administrator the available services. If the description is not sufficient the tool asks the administrator.

(P5) public services must be accessible from the Internet.

This policy states that Internet users to use public services, typically hosted in a DMZ. Consequently the administrator should identify which are the public services (reading the related documentation or analyzing network) and their properties (protocols, IP addresses). For example, considering a web portal, the administrator should configure the border firewall to allow HTTP(s) traffic from Internet to web server that hosts the service. Finally, to translate this information into real configurations it is necessary to consider the type of device. Several firewalls don’t support layer 7 but only layer 4 filtering, and in that case, the policy information are translated allowing Internet traffic to web server on 80/443 ports and TCP protocol. The automatic tool, using a set of reasonings, derives the public services (from network and zone analysis) and asks the administrator for confirmation. Afterward the tool: 1) identifies the firewalls involved among public services and Internet; 2) generates the related configurations.

(P6) guest users have to access the Internet resources.

Several organizations isolate visiting users from the private network in a particular sub-network but allow them to access the Internet. In that case, the administrator configures the involved filtering devices to allow traffic from the guest network address to Internet. Analogously, the tool relies on the tagging of guest zone in the network description. If this information is not available the tool asks the administrator.

(P7) the organization’s business services have to access their business components.

This policy and the next one (P8) are concerned with interactions among the service components which form a business service. For each business service the administrator has to evaluate the interactions between its components (through their advertised SAPs) and the functional and security properties required for these interactions. Sometimes the components are displaced in different networks with distinct security levels. In that case, the administrator identifies which filtering devices should be configured by defining the set of rules to allow traffic among service components. The tool relies on business service descriptions which should be provided using a language similar to WSCDL in order to highlight components and interactions.

(P8) business services must be connected securely to their components located in other parts of the network at higher security.

This policy concerns the protection of the confidentiality and integrity of the entire business services. In case of different security levels due physical displacement, the administrator could adopt different channel protection technologies (e.g., IPsec, SSL/TLS). The description of business services must be detailed enough to contain a set of security properties in order to identify the type of protection for each communication. The tool analyses the channel protection technologies available on services that should communicate (e.g., SSL/TLS). In some cases, when these technologies are not available on services, the tool evaluates the channel protection functionalities of the neighbour devices (e.g. IPsec).

4.3 Knowledge Management Area

The process of policy translation has been designed to mimic the behaviour of skilled and experienced administrators in acquiring environmental information and translating it to actual rules enforcing best security practices.

In general, the task performed by the administrator when configuring the policy in a given networked environment can be abstracted by five phases:

- *Policy analysis and understanding*, the administrator reads the company goals and associates them to basic security properties. For example, for policy P1 the administrator understands that the category of controls to configure are the filtering devices while for policy P8 he understands that he must activate some channel protection facility.
- *Entity identification*, the communicating network entities affected by the policy are identified. At this level, the policy is well represented using logical associations between entities, e.g., (users, have-to-access, services).
- *Entity characterization*, all the necessary low level properties regarding the identified properties are gathered, such as IP addresses, listening ports of the services, protocols, and available cryptographic algorithms.
- *Security enabled device and end-nodes identification*, the devices and the end-nodes present in the information system that are actually able to enforce the policy are determined.
- *Security-enabled devices selection and configuration*, the devices to actually configure are chosen and the ACLs (rules sets) to configure each of them are derived.

Thus, according to the RFC 3198, the policy translation can be seen as a progressive knowledge enrichment and continuous specification of the concepts that are expressed by means of the policy towards device-level parameters. Most of these steps require to reason upon the information about the system to configure. For instance, in the examples presented before, public services are a case of "services he is authorized to use" and therefore an administrator must be able to identify them. Furthermore, the "security levels" need to be derived in order to give meaning to the sentence "other parts of the network at higher security".

The initial ontology contains the formal representation of the set of concepts and objects that characterize our domain of interest, that is: 1) the system to configure; 2) the security properties that need to be enforced; 3) the relationships between those concepts (properties and logical inferences).

OPoT performs knowledge enrichment through three entities: an ontology reasoner, the Reasoning Manager and the Configuration Manager.

The ontology reasoners are tools that perform reasoning services to the ontology. For instance, it uses assertions, applies properties (e.g., transitive, symmetric properties, etc.), derives sub-properties, performs consistency checking (cardinality, user-define data ranges, etc.), concepts satisfiability, classifies individuals by analysing sub-classes relations, and the realization, that is, it computes the direct types for each individual.

The Reasoning Manager coordinates a set of independent modules called "Reasonings" that apply logical inference on the knowledge base to increment it that is not directly obtainable through the standard ontology reasoner. Example of reasonings are the automatic identification of DMZs, the classification of services according to their scope, the classification of SAPs according to their channel protection capabilities, or the identification of the security levels of the network zones.

However, not all the information required to actually configure the security-enabled element can be derived from the provided inputs. The Configuration Manager coordinates the import of all the non-deductible information. In particular, the Configuration Manager drives the administrator in collecting all the missing external environmental information that is not actually obtainable using the available reasonings. The administrator is asked to provide necessary through the Configuration GUI in a structured manner. Additionally, hints and predefined answers are presented in order to reduce errors. For example, in our proof of concept the administrator may be asked to provide the host-user assignments, the user's task assignment (to understand the services he must access) expressed according to company's organization roles, or the description of business services provided through the network (e.g., using the WS-CDL format).

4.4 Ontology

Our security ontology is structured in three levels (Figure 4).

The first level contains classes whose instances cannot be derived from other classes. First level instances are gathered using information contained in external files and algorithms to aggregate it (e.g., the filtering zones).

The next levels are generated through a reasoning process, that is, using the ontology reasoner or through the different modules called by the Reasoning Manager. The higher the level the more the information is detailed, e.g., in the second level computers are classified in workstations or servers and in the third level the workstations are further classified in shared or personal. The security ontology contains several properties linking the instances of different classes. This permits to navigate and analyze the ontology for deduction purposes. Thus, for example, an instance in the server class has a property that permits linking it to the instances of the SAP and it is linked to the Ciphersuites.

Figure 4 displays an extract of the class hierarchy associated to our security ontology.

4.5 Output

Finally, the Output Unit derives the logical associations from the ontology thus completing the translation. A logical association expresses an interaction between parties (typically one-to-many) and the related communication properties (e.g., filtering constraints). But it is not our desired output, yet. The translation process distinguishes between topological-independent and topological-dependent logical associations. In the first case (e.g., end-to-end interactions such as SSL/TLS protected channels) the process directly generates the configurations for the end nodes. In the second case, the network topology needs to be analyzed. All paths between the source and destination nodes are identified and rules are generated for the involved

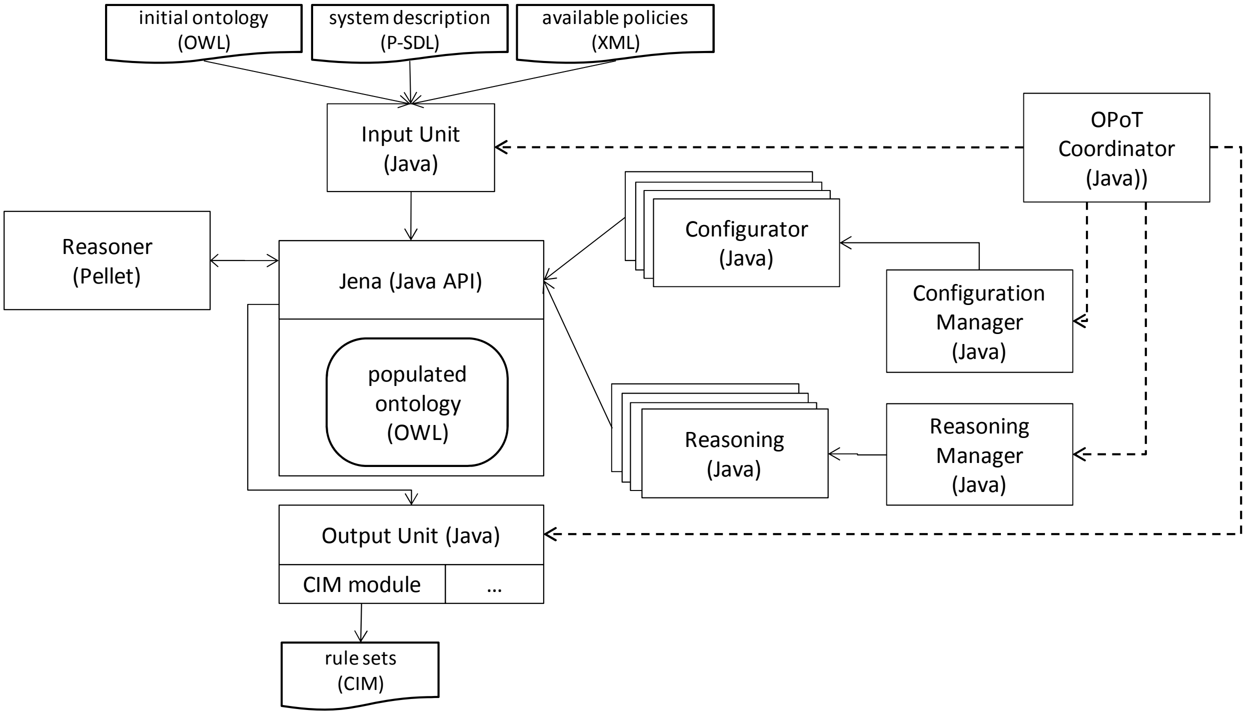


Figure 5. The prototype implementation.

devices. For example, when a company client needs to reach an internal service, OPoT configures all the firewalls in the paths between them. The network contextualization process is well described in this paper [16].

The presence of an output unit permits to have a great flexibility. Presently, OPoT generates the configurations using a format inspired to the CIM Simplified Policy Language [17]. More precisely, our output format is a use profile of the IETF Policy Core Information Model [18] and its extensions [19]. Nevertheless, other formats may be easily inserted by developing ad-hoc modules.

4.6 Workflow

The logical workflow of our framework (Figure 3a) is quite simple. First of all, the available policies are read from a configuration file, and then the administrator is prompted to choose among them which he wants to enforce. After this, the system description is retrieved from a configuration file and used to populate the initial ontology. The ontology reasoner is used for the first classifications and inferences.

Every policy entails the type of information required to refine it, that is, the modules selected by Reasoning Manager and the data to be acquired by the Configuration Manager is determined by the set of chosen policies. Therefore, the OPoT coordinator runs the Reasoning Manager that executes the implied reasonings. Reasonings enrich the knowledge but they may also warn the administrator about possible anomalies.

If some of the information needed to translate one of the chosen policies is not derivable using the reasonings, the Configuration Manager asks the administrator to provide missing data through the Configuration GUI. The Reasoning Manager is run before the Configuration Manager because it does not require any effort from the administrators thus simplifying the entire process. If some information obtained by means of the Configuration Manager enables other reasonings, the control returns to the Reasoning Manager.

This cycle may run more than once until the policy is enforceable. If in a cycle the ontology is not modified but the needed information is not present, the policy is considered not enforceable and a warning message is sent to the administrator that may change the chosen policies.

5. An example of policy translation

As an example of application of OPoT to policy translation we present the case of derivation of filtering rules associated to the policy “Every user must access only the services he is authorized to use”. This represents a typical security and management goal in many organizations, implicitly adopted during network configuration. Translating this policy means, as a minimum, to derive associations of the following type, to be enforced on filtering devices or application-level filters:

$$(IP_{source}; port_{source}; IP_{dest}; port_{dest}. proto)$$

The main effort for translating this policy consists in identifying users and services, categorizing the services according to their scope (e.g., public, private, only for a specific sub-network) then mapping them to IP addresses and port number.

Service descriptions are present in the ontology in the form of SAPs obtained from the system description.

The association between users and IP addresses is a typical piece of information that cannot be derived by the system description. For this reason, OPoT asks the administrator to provide information about the workstation assignment: a user is assumed to use his workstation or the shared ones. The workstations are automatically identified as the computer systems exposing no services.

To define the associations, OPoT has been designed to apply the best practice stating that a user is supposed to be authorized to access:

- all the private services needed for performing his work;
- all the company's public services;
- all the services that sub-net/zone exposes to the rest of the network;
- the Internet.

This best practice is inspired to the "Remote Access Policy" [15]. The concepts that need to be translated are "users", "services", in particular, "public services", "private services", "Internet" and the "users".

The first step is to classify services. As a pre-requisite the DMZ must be identified. The objective of a DMZ is to provide and group services to a zone considered at a lower security level. The DMZs are automatically identified using the ontology reasoner. In case of doubt, the ExpectedDMZ class is used. Usually a public DMZ is a sub-network directly connected to the border firewall (if there is one) or to a firewall connected to a border router where services are located. Other internal DMZs are present in company's networks to separate different portions of the network according to the security level. A typical example is those services that the administration unit makes available to the rest of the company. For example, in our institution the Student Management sub-network exposes through a DMZ the services to show students their academic data, print official documents, and get certificates, and to permit instructors to register the examination grades.

Additionally classifying network zones according to the security levels is useful for the identification of the scope of services. We discuss here the first component of the security level, that is, the number of encountered DMZ. The Internet is supposed to be at level 0, while the internal network is at least at level 1. The global DMZ is classified at level 1 too. Starting from this foundation, the reasoning looks for other internal DMZs, identifies the separation domains, and assigns the first component of the security levels accordingly. The assumption in translating the policy is that a service hosted in a DMZ at level L must be available to zones at level $L–1$ but not to the ones at the level $L– 2$. For instance, in Figure 1, it is possible to see the DMZ 1 where the public services are exposed to the Internet. Thus, the security level of the "internal" zones is at least one. Due to the presence of the DMZ 2, the Zones 6 and 7 are classified at level 2. For services for which it is not possible to identify the scope, administrator is asked to explicitly provide the classification. See Figure 1 to view the complete network classification according to the security level.

After this phase, two types of anomalies are reported to the administrators: the DMZ should not contain workstations, and the services in zones at level L should not be accessible to hosts in zones at level $L–2$ or smaller (because this creates a breach in the security compartments). For example, the DMZ 1 contains one workstation (see Figure 2). When all the services are classified, it is easy to get from the system description the IP addresses and ports.

The remaining task is to associate users to network entities and their duties inside the company. Since in principle it is not possible to deduce which services a user needs to work, the Configuration Manager asks administrator to include the explicit user-role RBAC description also containing the remote access policy. A particular attention is devoted to shared workstation. They are nodes available to different types of users and therefore must be able to access all the services needed by these users to perform their work.

At the end, all the needed logical associations have been derived as ontological properties to be further translated in ACLs.

6. Implementation

OPoT is implemented using the Java language (Figure 5). The Jena API [20] is used to manage the security ontology. At present, OPoT supports the P-SDL language [21]. P-SDL is a language defined to provide a format for description of networked ICT systems whose main purpose is to provide the data needed to perform various kind of security analysis. The main elements of P-SDL are physical elements, like hosts, devices and hardware platforms and the different network connections between these elements. On top of the physical network layer, it is possible to model logical elements like software components and services.

This prototype receives the environmental data written in XML. We used standards when available (e.g., the RBAC profile of XACML [22]). Our security ontology is written in OWL-DL [23] using Protégé [24] as editor. A set of classes parses the system description and runs a series of graph-based algorithms to extract additional information (e.g., the filtering zones). The policies are represented as Java classes. A policy links the reasonings to gather the information needed to its translation. Also reasonings are implemented as Java classes that interact with the ontology through the Jena framework and the Pellet reasoner [25] [26].

While deriving the logical associations, OPoT performs several controls, according to best practice, to find ambiguities or conflicts. In case of anomalies, it shows to administrator the list of the problems and suggests him the possible solutions. Finally, the logical associations are extracted using SPARQL [27] to query ontology. Considering the network in Figure 5 and seven out of eight policies, the tool uses about two minutes of CPU time from the policy selection to the translation. This test was performed using a PC with 2 GHz CPU and 1 GB of RAM. The max memory utilization was 30 MB.

7. Conclusions and future work

This paper presented OPoT, an ontology-based approach that addresses the problem of policy translation. We developed a security ontology to drive the administrator from policy specification to system configuration and a tool to drive administrators in the difficult task of refining policies into device configurations. The OPoT system can be extended to use other access control mechanisms, such as IEEE 802.1x. Moreover, automatic reasoning is not currently able to cope with the derivation of policies from automatic lexical analysis of texts. In future, when the semantic analysis of text will came of age, it would be possible to derive the policies from the textual specification then they will be mapped on the translation process.

References

[1] K. Kark. “How to Manage Your Information Security Policy Framework”. *Forrester Research*, 2006.

[2] A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, S. Waldbusser. “Terminology for Policy-Based Management”. *RFC 3198*, IETF, 2001.

[3] J. Strassner. *Policy Based Network Management*, Morgan Kauffman Publishers, 2004.

[4] T. R. Gruber. “Toward Principles for the Design of Ontologies Used for Knowledge Sharing”, *International Journal Human-Computer Studies*, 43 (5-6), pp. 907-928, 1995.

[5] J. Strassner, J. Neuman de Souza, D. Raymer, S. Samudrala, S. Davy, K. Barrett. “The design of a new policy model to support ontology-driven reasoning for autonomic networking”. In *5th Latin American Network Operations and Management Symposium (LANOMS)*, pp. 114-125, 2007.

[6] B. Tsoumas, D. Gritzalis. “Towards an ontology-based security management”. In *20th International Conference on Advanced Information Networking and Applications*, pp. 985-992, 2006.

[7] S. Fenz, A. Ekelhart. “Formalizing information security knowledge”. In *4th International Symposium on Information, Computer, and Communications Security (ASIACCS ’09)*, pp. 183-194, 2009.

[8] A. Ekelhart, S. Fenz, M. Klemen, E. Weippl. “Security ontologies: Improving Quantitative Risk Analysis”. In *40th Annual Hawaii International Conference on System Sciences (HICSS 2007)*, pp. 156a-156a, 2007.

[9] A. Uszok, J. M. Bradshaw, M. Johnson, R. Jeffers, A. Tate, J. Dalton, S. Aitken. “KAoS policy management for semantic Web services”, *IEEE Intelligent Systems Magazine*, 19 (4), pp. 32-41, 2004.

[10] A. Uszok, J. Bradshaw, J. Lott, et al. “New Developments in Ontology-Based Policy Management: Increasing the Practicality and Comprehensiveness of KAoS”. In *IEEE Workshop on Policies for Distributed Systems and Networks (POLICY 2008)*, pp. 145-152, 2008.

[11] E. Al-Shaer, H. Hamed. “Modeling and Management of Firewall Policies”, *IEEE Transactions on Network and Service Management*, 1 (1), pp. 2-10, 2004.

[12] A. Mayer, A. Wool, E. Ziskind. “Offline firewall analysis”, *International Journal of Information Security*, 5 (3), pp. 125-144, 2006.

[13] R. Ferraiolo, R. Kacker, R. Kuhn. “The NIST Model for Role Based Access Control: Toward a Unified Standard”. In *5th ACM Workshop on Role Based Access Control*, pp.47-63, 2000.

[14] P. A. Loscocco, S. D. Smalley, P. A. Muckelbauer, R. C. Taylor, S. J. Turner, J. F. Farrell. “The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments”. In *21st National Information Systems Security Conference*, pp. 303-314, 1998.

[15] Various Authors. “The SANS Security Policy Project”. SANS, 2009.

[16] C. Basile, A. Lioy, M. Vallini. “Towards a network-independent policy specification”. In *PDP2010: 18th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, Pisa (Italy), Feb. 17-19, 2010.

[17] J. Lobo, “CIM Simplified policy language (CIM-SPL)”. DMTF *DSP0231*, Distributed Management Task Force, 2007.

[18] B. Moore, E. Ellesson, J. Strassner, A. Westerinen. “Policy core information model”. *RFC 3060*, IETF, 2001.

[19] B. Moore. “Policy core information model (PCIM) extensions”. *RFC 3460*, IETF, 2003.

[20] HP-Labs. “Jena a semantic web framework for java”. 2009, http://jena.sourceforge.net/.

[21] POSITIF Consortium. “The POSITIF system description language (P-SDL)”. 2007, http://security.polito.it/positif/.

[22] A. Anderson. “Core and hierarchical role based access control (RBAC) profile of XACML v2.0”. *OASIS standard, 1 February 2005*, OASIS, 2005.

[23] W3C OWL Working Group. “OWL 2 Web Ontology Language”. *W3C Recommendation 27 October 2009*.

[24] Stanford University. “Protégé”. 2009, http://protege.stanford.edu/.

[25] Clark&Parsia. “Pellet: The Open Source OWL DL Reasoner”. 2009, http://clarkparsia.com/pellet.

[26] S. Bechhofer, R. Möller, P. Crowther. “The DIG Description Logic Interface”. In *Proceedings of the 2003 Description Logic Workshop (DL2003)*, 2003.

[27] K. G. Clark, L. Feigenbaum, E. Torres. “SPARQL Protocol for RDF”. *W3C Recommendation 15 January 2008*, 2008.

Author Biographies

Cataldo Basile holds a M.Sc. and a Ph.D. in Computer Engineering from the Politecnico di Torino where is currently a research assistant. His research is concerned with policy-based management of security in networked environments, automatic refinement of policies to device configurations and general models for detection, resolution and reconciliation of specification conflicts.

Antonio Lioy holds a M.Sc. in Electronic Engineering (*summa cum laude*) and a Ph.D. in Computer Engineering, both from Politecnico di Torino, Italy. He is currently Full Professor of Computer Engineering at the Politecnico di Torino where he leads the TORSEC security group. His research interests are in the area of policy-based security, network security and PKI.

Salvatore Scozzi holds a M.Sc. in Computer Engineering from the Politecnico di Torino where is currently a research assistant. His research activities regard ontologies and their application to automatic refinement of security policies and the security analysis of an ICT system.

Marco Vallini holds a M.Sc. in Computer Engineering from the Politecnico di Torino where is currently a PhD student. His research activities regard policy-based network management and automatic refinement of security and dependability mechanisms.