Application of Ontologies and Formal Behaviour Definitions for Automated Intrusion Response Systems

Verónica Mateos, Víctor A. Villagrá and Julio Berrocal
Dpto. de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid, Madrid, Spain
Emails: {vmateos, villagra, berrocal}@dit.upm.es

Automated intrusion reaction is an important problem in network security nowadays, because just intrusion detection is usually not enough to manage the proliferation of the attacks in networks. The existing AIRS use different mechanisms to infer the optimum response when a security incident is detected, but there is no standardized framework that models all the elements used in the reaction process. This paper proposes an ontology-based approach for automatically responding against intrusions. An intrusion response ontology which formally defines all the information needed in the intrusion response process is defined, and inference rules are specified using formal behaviour languages. Finally, some results are shown concerning the evaluation of this approach in a Denial of Service use case.

Keywords: *intrusion response, security ontology, OWL, SWRL*

ACM Classifications: *C.2.0 Computer-Communication Networks: Security and protection; D.2.8 Software Engineering: Metrics; D.2.11 Software Engineering: Software Architectures – domain-specific architectures; D.3.1 Programming Languages: Formal Definitions and Theory – semantics; D.4.6 Operating Systems: Security and Protection – Invasive software; F.4.3 Mathematical Logic and Formal Languages: Formal Languages; I.2.3 Artificial Intelligence: Deduction and Theorem Providing – answer/reason extraction, inference engines; I.2.4 Artificial Intelligence: Knowlegde Representation Formalisms and Methods – predicate logic, representation languages, semantic networks; K.6.5 Management of Computing and Information Systems: Security and Protection – invasive software, unauthorized access*

General Terms: *Design, Languages, Security*

1. Introduction

Automated intrusion reaction is an area of great interest currently, because just intrusion detection is usually not enough to manage the proliferation of the attacks in networks nowadays. Security incidents in networks are more and more sophisticated, intrusive and quickly spread, and the reaction time between the IDSs detecting the intrusion and the administrator performing the suitable response action can be really high, which can cause irreparable damage to the compromised systems. For this reason, in recent years, several AIRSs (Automated Intrusion Response Systems) have been proposed, in order to automatically react against the intrusions, such as that proposed by Stakhanova *et al* (2007–1), ADEPTS (Wu *et al*, 2007), FAIR (Papadaki and Furnell, 2006), AAIRS (Carver, 2001), etc. These systems use different mechanisms to infer the optimum response or responses when a security incident is detected, but there is no standardized framework that

Manuscript received: 1 November 2012
Communicating Editor: Rafael Valencia-Garcia

models all the elements used in the reaction process. The optimum response refers to the most suitable reaction against a particular intrusion in a specific moment, whose aim is to mitigate this attack or reduce its impact.

Also, in a collaborative network, where different detection sources send alerts to an AIRS using different formats and syntax, and the output of the AIRS can be used by other response systems in collaborative networks, it should be advisable to use a standard information model that enables a common understanding of the alerts, intrusions, or responses for all the elements involved in the intrusion reaction process.

Ontologies and semantic web technologies provide the means to include semantic awareness in the information they define. These technologies have proved to be really useful to allow different and heterogeneous agents to share and reuse knowledge, in order to solve heterogeneity problems that arose between different modeling languages of information and knowledge. Ontologies are used to map and merge different intrusion alert models, taking into account their semantics in a common ontology-based model. Furthermore, there are behaviour languages and semantic reasoners related to ontologies that allow systems to formally specify their behaviour, and based on that, infer and learn new knowledge, which is another advantage for using ontologies and semantic web technologies for getting the optimum reaction against an intrusion.

This paper proposes an ontology to formally define all the information and behaviour needed in the intrusion response process carried out by an AIRS. The architecture of an ontology-based AIRS is presented in a previous work of the authors (Mateos et al, 2010). This paper provides the complete definition and analysis of the intrusion response ontology and inference rules that was introduced in that previous paper.

The paper is organized as follows: Section 2 provides a discussion of the proposal in relation to existing work in this area. Section 3 gives an overview on the technical framework. Section 4 describes the intrusion response ontology proposed and how to use this ontology, behaviour languages and semantic reasoners to infer and learn new knowledge in an AIRS. Section 5 presents an actual case that applies this work to an intrusion scenario. Finally, Section 6 concludes the paper and provides a discussion about the future work in this area.

2. Related Work

The use of taxonomies and ontologies in network security is not new. In recent years, a lot of proposals have been done related to how to classify and model security information. Opposite to ontologies, taxonomies lack the necessary and sufficient constructs needed to reason with instances of the modeled domain.

In Souag *et al* (2012), the authors review, analyze, select and classify security ontologies, as an overview of what exists in terms of security ontologies. They classify existent ontologies in eight families: beginning security ontologies, security taxonomies, general security ontologies, specific security ontologies, web oriented security ontologies, risk based security ontologies, ontologies for security requirements and security modelling ontologies. Undercoffer *et al* (2003) present a specific security ontology for intrusion detection specifying a model of computer attack using DAML+OIL. This ontology models concepts related to intrusion, such as Attack, System Component, Host, or Consequence.

In Vorobiev and Han (2006), a security attack ontology is proposed for Web Services, easy to implement and deploy to protect WS based system from different types of attacks including

distributed and multi-phased attacks. The ontology includes the definition of several attack classes such as, WS DoS, CDATA Field, Application Attacks, SOAP attacks, etc.

Tsoumas and Gritzalis (2006) define an ontology for security requirements. They define a standards-based knowledge ontology which extends the CIM model containing concepts relating to Risk Assessment, such as Asset, Stakeholder, Vulnerability, Countermeasure and Threat, in order to demonstrate that the security information extraction from high-level statements is feasible.

Abdoli and Kahani (2009) present an attack ontology for intrusion detection in distributed systems. The designed ontology has one main class, called "attack". This class contains all kinds of computer attacks and has so many subclasses. They propose to use this ontology in order to reduce the rate of false positive and false negative of the current IDSs.

Herzog *et al* (2007) define a general security ontology that tries to cover all security aspects, by containing detailed domain vocabulary and is thus capable of answering queries about specific, technical, security problems and solutions. They used OWL to model core concepts really significant in security: asset, threat, vulnerability, countermeasure, security goals and defense strategy. All the core concepts are subclassed or instantiated to provide the domain vocabulary of information security. Also they define relations to connect different concepts.

Besides the definition and specification of security ontologies, some works have been done to formally define behaviour of the intrusion detection, correlation and response systems.

Li and Tian (2008) propose an ontology-based system for intrusion alerts correlation. They define an alert correlation ontology frame used by a multi-agent system architecture consisting of agents and sensors. A State Sensor collects information about security state and a Local State Agent and Center State Agent preprocess the security state information and convert it to ontology. An Attack Sensor collects information about attack and the Local Alert Agent and Center Alert Agent preprocess the alert information and convert it to ontology. The Attack Correlator correlates the attacks and outputs the attack sessions. The ontology models concepts such as Attack, Attacker, Asset, System State, etc.

Lopez de Vergara *et al* (2009) propose an ontology-based methodology for the instantiation of new security policies that counteract the network attacks. The idea is mapping alerts into attack context, which are later used to identify the security policies to be applied in the network to solve the threat. For this, authors define two ontologies that represent IDMEF (Intrusion Detection Message Exchange Format) alerts and ORBAC (Organization Based Access Control) policies.

Finally, Azevedo *et al* (2010) propose an autonomic model called AutoCore constituted by a Multi-Agent system, an Intelligent Interface and a formal ontology (CoreSec) in order to address the problem of protecting computational resources, by managing computational security in corporative environments. The purpose of the system is to provide a formal model that reflects the security aspects for this type of environment, allowing a safe communication between several elements of the autonomic system and enabling a process of inference concerning the different events that can put the security of the systems at risk.

The analysis of these proposals shows that there are a lot of interesting research works around the security ontologies and the definition of formal behaviour of security systems, but on the other hand, it is really difficult to find an ontology to cover all the aspects related to this area.

The approach proposed in this paper is focused on a new problem: infer the optimum response to automatically react against a detected intrusion in a specific network. None of the analyzed

ontologies completely model the intrusion response process, but it is possible to partially reuse some of them for building our specific ontology. Besides the intrusion response ontology, the AIRS approach uses SWRL (Semantic Web Rule Language) and semantic reasoners to infer and learn new knowledge.

3 Technology Overview

3.1 Ontologies: OWL and SWRL

Ontologies are one of the main approaches used in the scope of Knowledge Management and Semantic Web to solve questions related to semantics, such as formally representing a set of concepts, their meaning and the interrelation between them. Formally, an ontology is "an explicit and formal specification of a shared conceptualization" (Guarino, 1998). Due to its great expressivity, nowadays, ontologies are used in many areas, such as security in communication networks, as mentioned in the previous section.

One of the main advantages of using ontologies for information modelling is that their semantics have been formalized. This is very relevant in environments where several information models are used, and some elements have to deal with definitions expressed in different information models. In this case, this element may not be able to recognize the same concept defined in different information models, leading to a variety of behaviour coding, one for each information model in which the concept is expressed. The use of ontologies provides a way to simplify that problem: an ontology has formalized the semantic aspects inside the definition of the concepts, and it is possible to find some method to match automatically or semiautomatically concepts defined in heterogeneous information models to the concepts defined in the ontology, such as the M&M (Merge and Map) method proposed in Lopez de Vergara *et al* (2010). This method is based on the merging method detailed in Noy and Musen (1999), but it is adapted to the case of the network and system management. In this particular work, the M&M method is adapted to the intrusion reaction area.

Within the scope of this work, using ontologies helps to support inclusion of different heterogeneous IDSs, with different intrusion formats and syntaxes. In this way, AIRS will be able to understand heterogeneous alerts and to know whether these alerts are referring to the same intrusion or not. Nowadays there are some data format standards for alerts representation, whose aim is to solve this problem, such as IDMEF (Debar *et al* (2007). This format defines a data model in the eXtensible Markup Language (XML) which allows the representation, the exchange and sharing of information about intrusion detection. But IDMEF only provides an exchange format, without any additional knowledge representation that can be useful for the AIRS in order to correlate this information with additional information, such as network context, rules, and so on.

Another great advantage of using ontologies for information modelling is the possibility to define also in the same framework integrated with the object information the behaviour of the defined objects. This behaviour definition can be expressed with a variety of artefacts (rules, logic, etc.) depending on the specific ontology and language chosen, but it will be integrated with the information definition, allowing to process and reason according to a predefined specification.

There are several ontology languages, such as KIF, Ontolingua, OCML, OKBC and F-Login (before the Semantic Web) and RDF, RDFS, DAML+OIL, OWL and OWL2 (XML-based). The main ontology languages used in the Semantic Web to formally describe information definitions are OWL (Smith *et al*, 2004) and OWL2. OWL is a knowledge definition language which structures the information into classes and properties (nominal or relation between objects), with hierarchies,

and range and domain restrictions. An OWL ontology contains a sequence of axioms and facts. It includes several types of axioms, such as subclass axioms, equivalent Class axioms and property constraints. In 2009 W3C announced the new version of OWL: OWL2, which introduces some enhancements in the knowledge representation aspect but still lacks the First Order Logic (FOL) needed to define rules.

The Semantic Web Rule Language (SWRL) (Horrocks *et al*, 2004) extends OWL with rule axioms. A rule axiom consists of an antecedent (body) and a consequent (head), each of which consists of a (possibly empty) set of atoms. In the "human-readable" syntax of SWRL, a rule has the form:

$$antecedent \Rightarrow consequent$$

Informally, a rule may be read as meaning that if the antecedent holds (is "true"), then the consequent must also hold. A SWRL rule has therefore the form of an implication relationship between the head and the body. The SWRL specification (Horrocks *et al*, 2004) provides an abstract syntax that extends the OWL abstract syntax to include this relationship in the ontology language.

Besides, SWRL allows including built-ins in the rules. SWRL built-ins are additional functions which make it possible to deal with XML Schema data types: mathematical operations, comparisons, logical negations, built-ins for Strings, built-ins for date, time and duration, URIs operations and built-ins for lists of elements.

In this work, OWL2 will be used as the ontology language to describe intrusion response concepts, and SWRL will be used as rule language to infer new knowledge.

3.2 Semantic Reasoner

A semantic reasoner, reasoning engine, rules engine, or simply a reasoner, is a piece of software able to infer logical consequences from a set of asserted facts or axioms. These axioms are provided by means of the OWL ontology and rule axioms (SWRL rules).

In the context of this work, the reasoner is the core of the AIRS, that is, the reasoner is the component of the architecture system which will infer the optimum response(s) against a detected intrusion.

Nowadays, there are several semantic reasoners that can be used in the AIRS, such as CEL, FaCT++, HermiT, Pellet, RacerPro, Bossam or REL. These reasoners can be classified according to several features: reasoning algorithm (rule based, tableau or Hyper tableau), SWRL rule support, consistency, incremental classification (addition/removal of new knowledge), ABox reasoning support, OWL API or Jena support. In this proposed work, SWRL is used as rule language, so, the semantic reasoner must support SWRL. For this reason, all reasoners except Hermit, Pellet, and Bossam are discarded. Besides, Hermit does not support SWRL built-in functions at all.

On the other hand, the total time of inference including ontology load time and reasoning time, is higher using Bossam than using Pellet. The tests and experiments done for getting this conclusion are beyond the scope of this paper.

Then, Pellet is most suitable for us to perform SWRL reasoning task based on the ontology, and the last version of Pellet is used as semantic reasoner of the AIRS.

3.3 AIRS

AIRSs are security technologies which trigger a dynamic reaction against a detected intrusion. The system infers the suitable response and triggers it automatically without needing the system administrator's participation.

In addition to this main functionality, there are additional features that AIRSs should include (Stakhanova *et al*, 2007-2): adaptability (AIRS must be able to modify the chosen optimum response according to the previous response effectiveness, the changes of the environment, etc.), sensitive to responses (system must take into account the complexity and cost of the reaction), and proactive (AIRS must be able to react against an intrusion before that intrusion takes place). But, there is another feature that is very important in a heterogeneous intrusion detection environment: semantic coherence. The semantic coherence feature is relevant: the intrusion response system would understand intrusion notifications with different syntaxes from different IDSs, and would be able to determine whether the two notifications refer semantically to the same intrusion or to different ones.

In recent years, several AIRS have been proposed, for example, ADEPTS (Adaptive Intrusion Tolerant Systems) (Wu *et al*, 2007), AAIRS (Adaptive Agent-Based Intrusion Response System) (Carver, 2001), AIRS proposed by Stakhanova *et al* (2007-1), and FAIR (Papadaki and Furnell, 2006), but none of these AIRS provide mechanisms to achieve semantic coherence.

The usage of ontologies, formal behaviour specification languages and reasoning mechanisms in the area of an AIRS solves the problem of semantic coherence. Moreover, due to its great expressiveness and flexibility, these technologies enable AIRS to meet other requirements: adaptability, proactivity and response cost sensitivity.

4. An Ontology for Intrusion Reaction

4.1 AIRS Architecture

To automatically react against a detected intrusion, an AIRS architecture is proposed, which is shown in the following figure (Mateos *et al*, 2010):

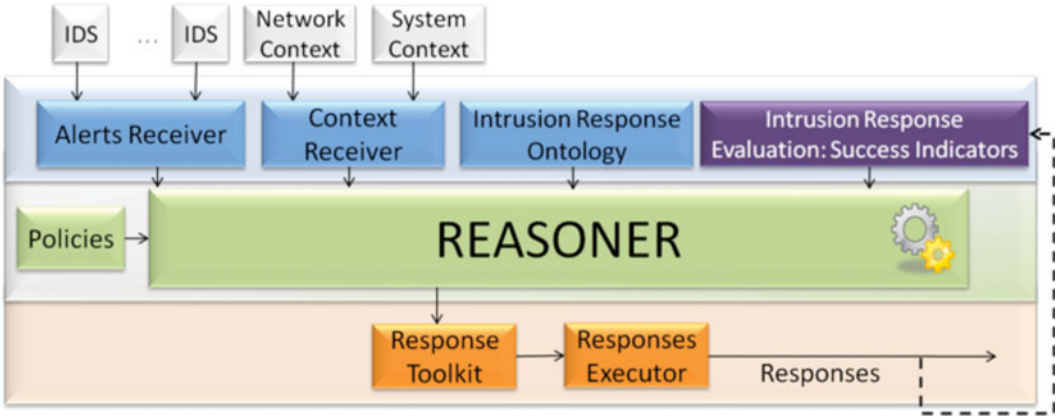

Figure 1: Architecture of the ontology-based AIRS

Three major phases have been defined in order to build the previous architecture:

Phase 1: Building the Intrusion Response Ontology

- First, OWL will be used as the ontology language in order to describe the intrusion response ontology, which is specified in Section 4.2.
- Then, technical data concerning the network topology, system components installed in the network (servers, host, etc) are located through the use of network scanning tools. Then, this

information is set in the ontology, by generating ontology instances from infrastructure data.

- Also, the administrator models the possible reactions included into the response toolkit in OWL, generating instances.

This process is explained with details in Section 4.2.

Phase 2: Definition of security metrics and their specification using SWRL.

The reasoner will infer concrete intrusion reaction that apply to detected attack, taking into account the information related to the intrusion (such as intrusion type or intrusion impact), the attacker (such as source IP address or location of the attacker), the compromised resource (such as destination IP address or level of importance), the network context (such as system latency or anomaly of the network) and the individual reactions (such as response efficiency or response cost). The policies which guide this process are modeled by SWRL rules that specify security metrics. The definition of these SWRL rules is explained in Section 4.3.

Phase 3: Implementation of the architecture

In this phase, a prototype was developed including the reasoner and the rest of modules. This prototype was implemented using JAVA code, as well as Jena API. Section 5 provides additional details about the usage and performance of this prototype in a given Use Case.

After designing and implementing the ontology-based AIRS, the system is executed. The reaction to intrusion is divided into three steps:

Step 1: Inference of the optimum response

– Generation of the ontological representation of the alerts sent by the IDSs. When the Alert Receiver receives a new alert from the IDSs, this alert is modeled as an instance of the ontology. This middleware layer translates from the IDS specific format (such as syslog or alert full output format of snort IDS) to the OWL language by means of OWL and Jena API.

– Collection of information about network context and system context, and ontological representation of this information. When a new intrusion is detected, the Network context and System context modules get the current context of the compromised system and the network, by calculating the anomaly degree for these parameters: number of active processes, CPU usage, free space in disk, system latency, system status, number of users and number of zombie processes, network traffic, etc. This information is then modeled also as ontology instances in OWL.

– Inference of a set of recommended responses according to the policies and the ontology including the new instances of alert and context.

– Choosing the optimum response among the recommended ones, with an additional inference process which uses the policies and the information included in the ontology related to assets in the organization.

Step 2: Running the chosen action

The Response Executor module carries out the inferred response.

Step 3: Evaluation of the response success or response effectiveness, in order to adapt the following reactions to these success values

One of the parameters included in the definition of the policies is the response success or response efficiency. As is explained in Mateos *et al* (2012), the highest severity and highest efficiency metric uses the response success, besides others parameters, to get the higher severity response against a particular intrusion type, and infer the optimum response.

Every time that an intrusion has been detected, and the suitable response has been executed and finished, the Response Evaluation module implements a quantitative method whose aim is to calculate the success or efficiency of the response triggered by the ontologies-based AIRS, by capturing, analyzing and comparing network and system information before and after the execution of the response. Outcomes achieved by the evaluation system will be included and updated into the ontology by means of Jena. In this way, the Reasoner module will take this parameter into account in further inferences of the optimum response.

4.2 Intrusion Response Ontology

The proposed intrusion response ontology formally defines all the information needed in the intrusion response process carried out by an AIRS, such as intrusion, responses, network context, IDSs, etc. as well as the relationships among them.

The efficiency and accuracy of the ontology-based AIRS depends on the quality and completeness of the ontology. Because of that, it is necessary to include a discussion of the ontology construction process. The purpose of this process is to find the exhaustive definition for each concept and each relation of the ontology in Description Logic language (DL language). The methodology used to define and build the ontology consists of four steps (Gómez-Pérez *et al*, 2004):

- *Build glossary of terms:* Specify the domain and scope of the ontology, and identify all the concepts which are included on it and its main features. In this step, we got the pairs (Object, Property).
- *Build concept taxonomies:* Build the hierarchy of concepts by grouping the terms sharing the same properties. We got two main taxonomies: a taxonomy of attacks which enables us to model the attacks both syntactic and semantically, and a taxonomy of responses which is a model that categorizes responses according to different criteria.
- *Specify relation diagrams:* Identify the relations between the different objects and their properties. In this step, we got the tuples (Object, Relation, Object).
- *Build the ontology:* This step is the representation of all concepts, relations, properties and instances in a DL language, including the definition of classes and the class hierarchy, the definition of the properties of classes specifying cardinality and value type, and the definition of the domain and range of relations.

The intrusion response ontology has been built using Protégé tool and OWL DL language. Besides using existing attacks and responses taxonomies to build the ontology, we have taken into account the security ontologies analyzed in Section 2, and we have partially reused some of their classes: the Threat and SecurityGoal classes are reused from the general security ontology proposed by Herzog *et al* (2007); and the Asset class is similar to the Target class included in the IDMEF ontology proposed by Lopez de Vergara *et al* (2009).

The ontology classes and the relationships between them can be seen in Figure 2, which has been generated using OntoViz.

A short description of each of the classes could be the following:

- Network: Describes the computer network of the institution or organization.
- Context: Models the main parameters that describe the context at intrusion time. It is possible to distinguish between two kinds of context: network context and system context. The first one is related to traffic parameters such as IP source address, or protocol. The second one is

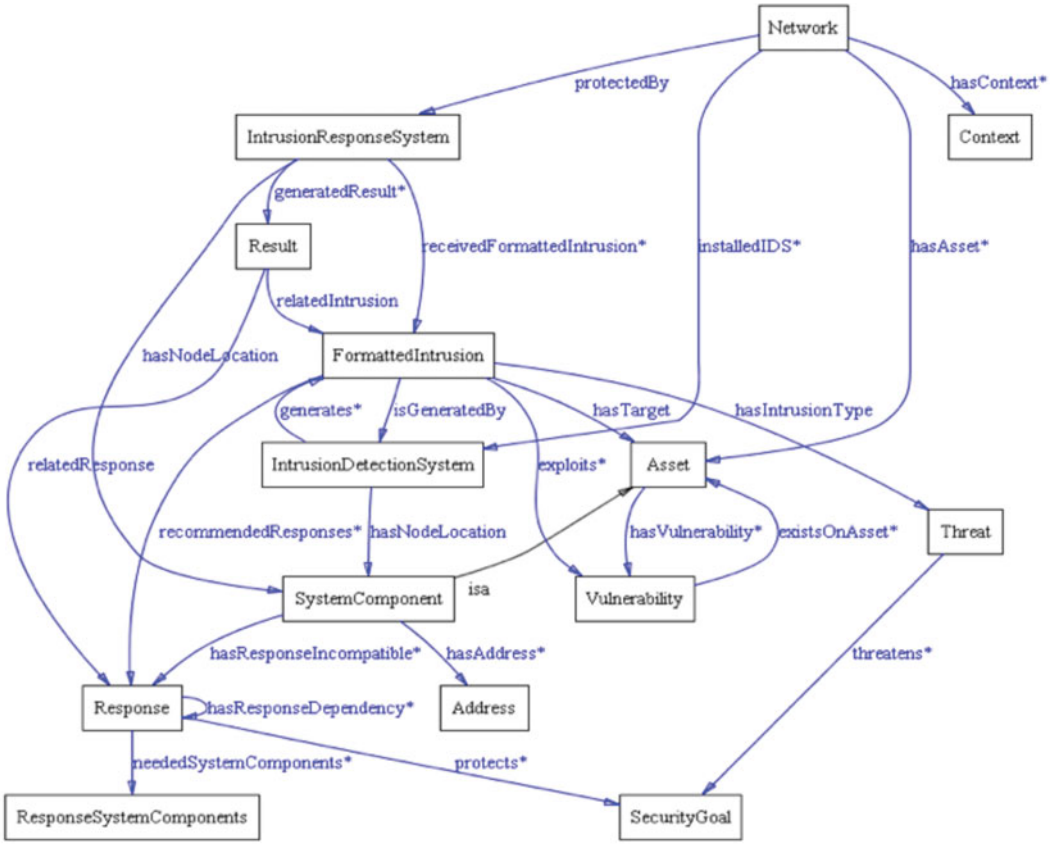

Figure 2: Intrusion Response Ontology

related to parameters of the compromised system, such as, number of active processes, CPU usage, etc.

- IntrusionDetectionSystem: Models the IDSs installed in the network. In this case, the network could have installed several IDSs.
- IntrusionResponseSystem: Models the network IRS. The network is protected only by an IRS, which receives the intrusion alerts (*receivedFormatedIntrusion* relation), infers the optimum response and generates the response application results (*generatedResult* relation).
- Asset: Models all the resources in the network that are necessary for the organization. This class has several subclasses including: User, SystemComponent or Node, Service, File and Proccess. The subclasses are not represented in the figure. These assets are the targets of the attacks (*hasTarget* relation).
- Address: Represents network address of a system component or an application. One system component or node can have more than one Address.
- Vulnerability: Models the system vulnerabilities. The exploitation of these vulnerabilities is the reason for the attack. Vulnerability is associated with one asset or more.
- Threat: Models the intrusion type. A *threat* threatens a SecurityGoal. Every FormattedIntrusion

has associated an intrusion type (*hastIntrusionType* relation).

- SecurityGoal: Specifies which security policy or policies are affected by the intrusion: confidentiality, integrity or availability.
- FormattedIntrusion: Represents the intrusion alerts generated by the IDSs. It is the most important ontology class along with the Response class. Each instance includes information about the intrusion type, attack source, intrusion impact, intrusion consequence, intrusion target asset, intrusion severity, timing of intrusion detection.
- Response: Models the responses that the AIRS can choose and execute, that is, the possible reactions included in the response toolkit. It is the most important ontology class along with FormattedIntrusion. The properties of this class define the different response features, such as: response cost, response complexity, response severity, etc.
- ResponseSystemComponents: Specifies the number of needed components to carry out a response as well as the number of available components.
- Result: Represents the evaluation result after executing a specific reaction. An instance of this class includes the intrusion identifier, the response identifier and response effectiveness or success.

Figures 3 and 4 depict the object properties related to FormattedIntrusion and Response respectively, because these are the most relevant classes of the ontology.

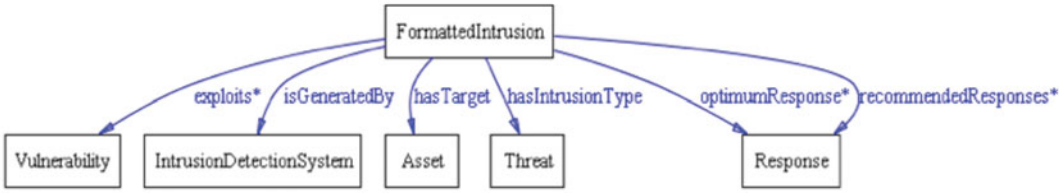

Figure 3: Intrusion Response Ontology (FormattedIntrusion relationships)

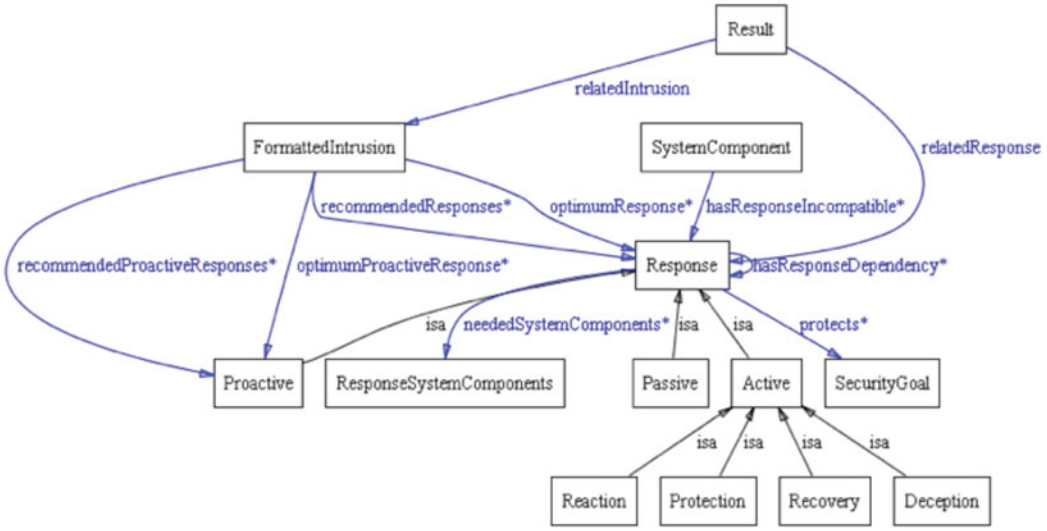

Figure 4: Intrusion Response Ontology (Response relationships)

The representation in a DL language is done to support reasoning (classification, instantiation and consistency checking). By means of a Protégé plug-in and Pellet, the consistency and the classification of the intrusion response ontology were successfully checked.

4.3 SWRL Rules Definition

Once the ontology has been defined, it is also important to define the policies or rules used by the inference engine. These policies are expressed using SWRL rules, and they specify the behaviour of the Reasoner module. Starting from these defined rules and the previous knowledge included in the ontology, the reasoner infers the most suitable response to a specific intrusion. SWRL rules also are suitable for specifying the response metrics for getting the optimum response.

Two set of rules have been defined: rules to infer the recommended responses and rules to choose the optimum response.

Rules to infer recommended responses

When an intrusion is detected, and the intrusion and acquired context information is modeled as new instances of the ontology, the reasoner obtains a set of recommended responses which are suitable for this intrusion according to several parameters, such as, intrusion type, and system and network context information.

This set is divided into three subsets:

- Rules for inferring proactive recommended responses.
- Rules for inferring reactive recommended responses (active or passive), if there are previous results against the same intrusion or similar ones. In this situation, rules check if there are results for the same intrusion or similar ones (attacks are never identical). The system considers that two intrusions are similar (almost identical) if the intrusion type, the resource compromised by the intrusion (type and IP) and the changes in the context at intrusion time are the same. If some previous result matches the rule and the related response was executed successfully, the system will select the same reaction. If not, the following subset of rules is executed.
- Rules for inferring reactive recommended responses if there are no previous results related to this intrusion or similar ones. For example, the following rule infers all the active responses that mitigate a threat of the same type as the detected intrusion, if the system latency is greater than 7, and the network anomaly is greater than 7, as recommended responses.

Active(?response),
hasIntrusionType(?intrusion, ?threat),
threatType(?threat, ?threattype),
orientedToThreatType(?response, ?resthreattype),
intrusionDetectionTime(?intrusion, ?intdate),
NetworkContext(?netcontext),
contextInformationDate(?netcontext, ?netdate),
SystemContext(?syscontext),
contextInformationDate(?syscontext, ?sysdate),
networkAnomaly(?netcontext, ?netan),
systemLatency(?syscontext, ?syslatency),
equal(?threattype, ?resthreattype),
equal(?netdate, ?intdate),

equal(?sysdate, ?intdate),
greaterThan(?netan, "7"^^int),
greaterThan(?syslatency, "7"^^int)
→ recommendedResponses(?intrusion, ?response)

Rules to infer optimum response

According to the set of recommended responses and the importance of the compromised asset, the AIRS chooses the optimum response against a specific intrusion. For doing that, three response metrics are proposed: damage reduction metric, minimum cost metric, and highest severity and highest efficiency metric. The equations of the proposed metrics are detailed in Mateos *et al* (2012), and they are specified by means of SWRL rules.

For example, the damage reduction metric satisfies an equation of the form:

$$Impact_{intrus} * Confidence_{IDS} \geq Impact_{respon} \quad (1)$$

This metric is specified with SWLR in the following way:

realintrusionImpact(?intrusion, ?intimpact),
responseImpact(?respon1, ?resimpact1),
swrlb:lessThanOrEqual(?resimpact1, ?intimpact)

where *realintrusionImpact* is the product of *intrusionImpact* and *IDSconfidence*; both are data type properties of the *FormattedIntrusion* class.

5. Use Case

This section provides an example based on the phases defined above and a use case devoted to a denial of service attack. The experiment evaluates the feasibility of using ontologies, rules languages and semantic reasoner for automatic reactions against an intrusion.

The organization network used can be seen in Figure 5. The scenario was comprised of several components:

- Three Linux (i386) hosts, which are not very important in the organization.
- A database server; this server is not publicly accessible and may contain sensitive information. It is a very important component.
- A DMZ web server, publicly accessible, which is more relevant than the hosts, but less than the database server.
- Snort IDS: The purpose of this system is to detect intrusions and send an alert to the AIRS using a syslog server listening in the udp port 514.
- Ontologies-based AIRS: This system includes the intrusion response ontology described in Section 4 and a set of rules that guide the AIRS behaviour. For the AIRS implementation, Pellet semantic reasoner has been used.

First of all, a set of instances has to be defined for modeling the system components installed in the network, and the possible reactions included into the response toolkit. Then, at intrusion time, three new instances need to be created related to the intrusion alert, the network context and system context. Finally, combining these instances with the SWRL rules in the Pellet inference engine, information about the recommended and optimum responses is obtained.

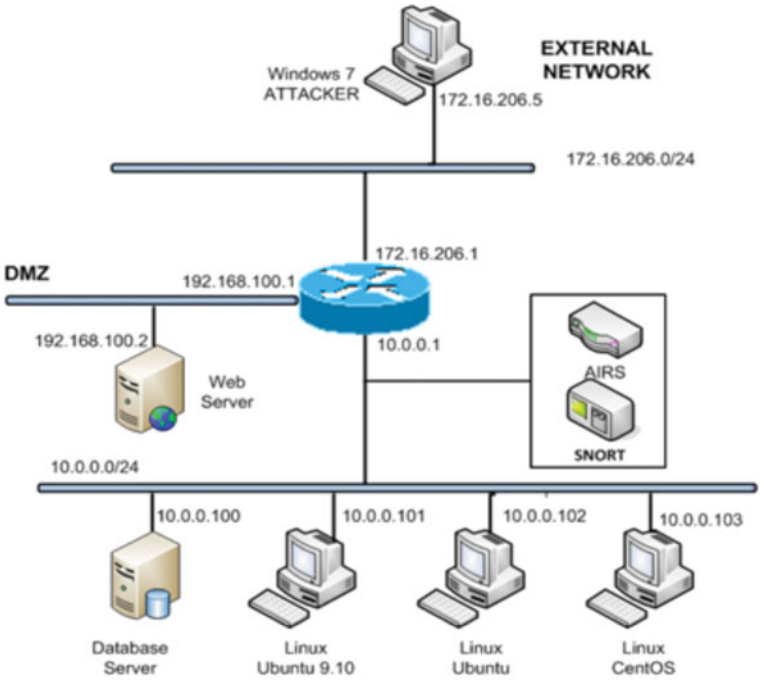

Figure 5: Use case organization network

5.1 Defining Initial Instances

The initial definition of instances for the Intrusion Response ontology includes Network, IDSs, IRS, Asset, and Response. A set of figures illustrate theses definitions, where red boxes are instances and io (Instance Of) black relationships indicate the instantiated class.

Scanning tools are used to obtain the topology or the network as well as the information related to assets or system components located in the organization. Then, the administrator generates ontology instances using a web interface that uses Jena for reading and writing into the ontology file, and JSP for presenting the results of the queries in a readable way. Figure 6 (on the following page) shows instances related to the organization network used in this use case. In this case, the network Network1 has six assets: INTNet-H1, INTNet-H2, INTNet-H3, INTNet-S1, INTNet-S2, and DMZ-S1. Also, the IDS1 and IRS1 are located in the system component INTNet-S2.

With respect to the description of responses, Figure 7 (on the following page) shows instances related to the different types of reactions that the AIRS is able to execute and the system component needed for executing them. For example, response Honeynet protects the confidentiality, integrity and availability security goals.

5.2 Adding Instances at Intrusion Detection Time

When the attacker throws a denial of service attack against the Web server of the organization, the IDSs will detect the intrusion, will generate an alert intrusion, and will send this alert to the ontology-based AIRS. Then, the AIRS will collect the anomaly of the network, and the number of active processes, CPU usage, free space in disk, system latency, system status, number of users and number of zombies processes, of the compromised asset.

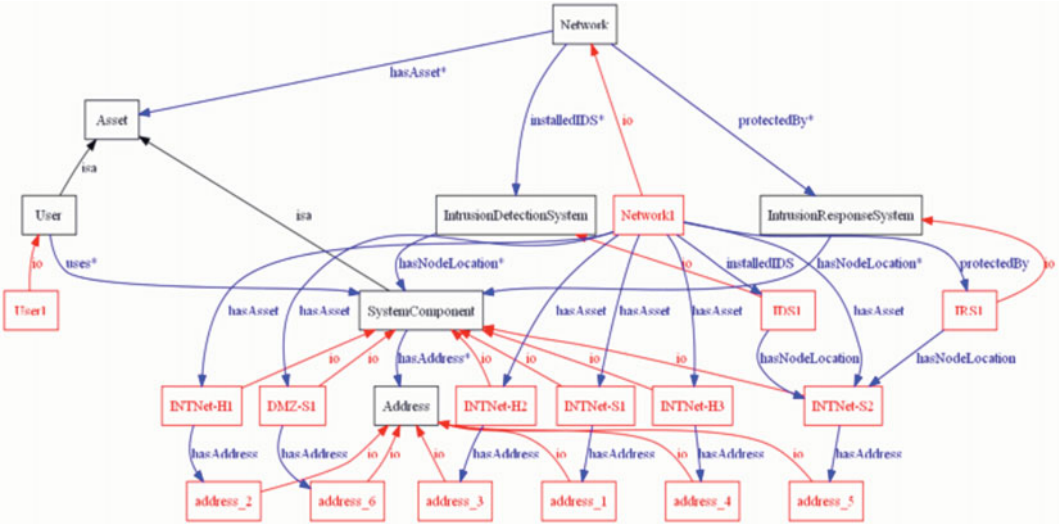

Figure 6: System component instances

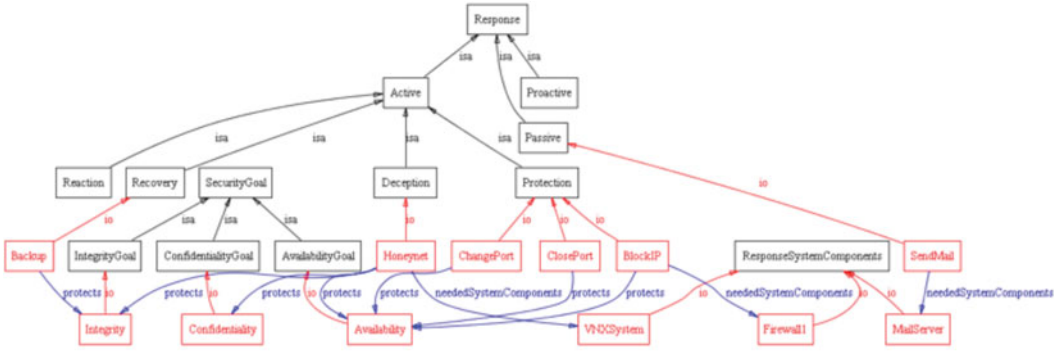

Figure 7: Response instances

Then, the AlertReceiver and ContextReceiver modules will map this information into the ontology by creating new instances, as is shown in Figures 8 and 9.

The new intrusion alert, intrusion2012-10-28T153745 is a Denial of Service attack detected by IDS1, whose target is DMZ-S1, and its impact is equal to 90.

The integer values of the SystemContext and NetworkContext instances are referred to the anomaly between the value at intrusion time and the value in a normal level of operation, of those parameters. For doing that, mathematical equations and models are used, but this is beyond the scope of this paper.

5.3 Inferring New Knowledge

Based on the set of instances defined above and SWRL rules, it is possible to know which are the recommended responses and the optimum response against a given attack. The following inferences are obtained:

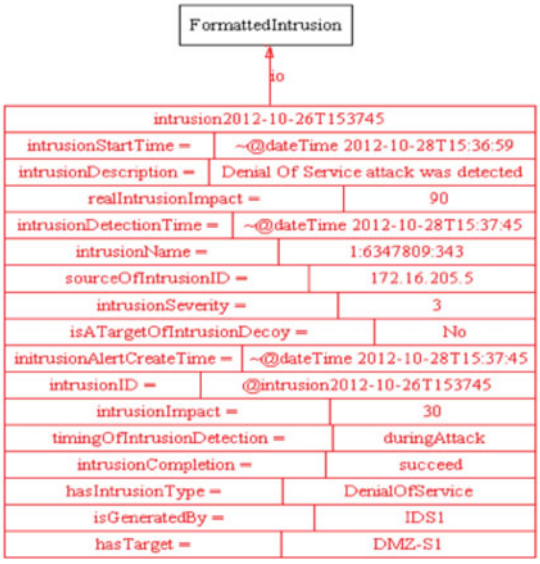

Figure 8: Created FormattedIntrusion instance (properties)

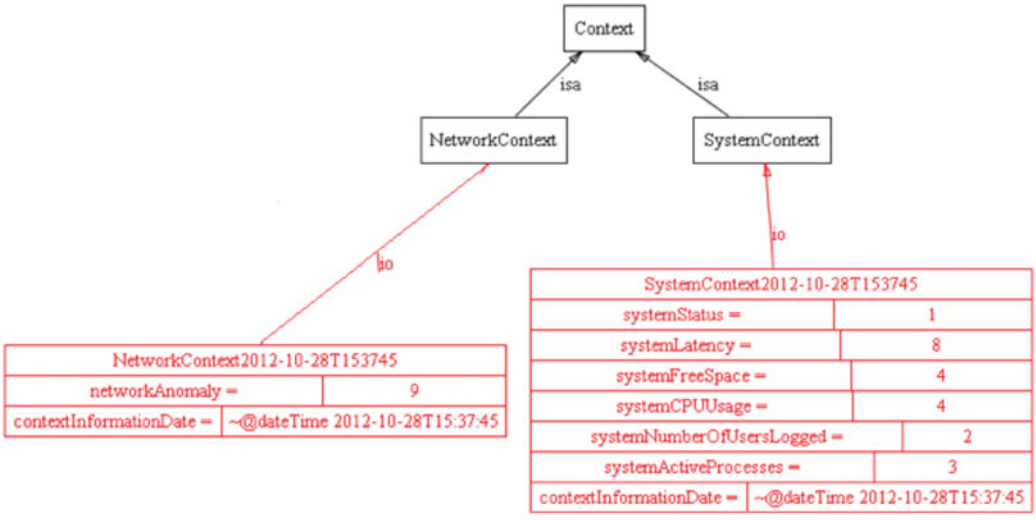

Figure 9: Created Context instances (properties)

1. Based on the data types properties of the system context, the networkAnomaly and the hasIntrusionType properties, SWRL rules identify the recommended responses suitable for a specific FormattedIntrusion. As a result, *BlockIP*, *ChangePort*, *ClosePort* and *SendMail* are recommended.
2. Based on the previous recommended responses and the assetLevelOfImportante property, a response metric will be used. In this step, SWRL rules include properties such as *intrusionImpact*, *responseImpact*, *responseCost*, *responseEfficiency*, *intrusionCost*, etc. In this case, the BlockIP response is inferred to mitigate the intrusion (see Figure 10).

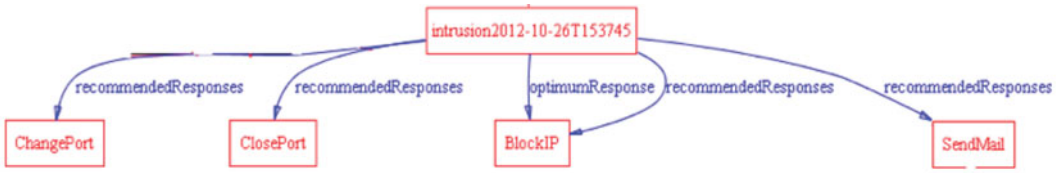

Figure 10: Recommended and Optimum responses inferred

Finally, in order to evaluate the performance of the AIRS, a set of tests have been done using the Intrusion Response ontology with the following number of components. Also, the total number of individuals, relations and concepts are specified:

- Network: 1
- IntrusionReactionSystem: 1
- User: 13
- Threat: 18
- ResponseSystemComponents: 4
- Address: 22
- SystemContext: 1
- SWRL rules: 32
- Total number of individuals: 96
- Number of different relations: 35
- Total number of relations: 169
- Number of different data type properties: 86
- Total number of data type properties: 418
- IntrusionDetectionSystem: 3
- SystemComponents: 20
- SecurityGoal: 3
- Response: 7
- Result: 0 or 1
- NetworkContext: 1
- FormattedIntrusion: 1

The results are presented in Table 1.

	No Previous Result	Successful Previous Result	Unsuccessful Previous Result
Ontology load time (ms)	250	250	250
Optimum Response inference time (ms)	255	205	200
Computing Time (ms)	7500	2000	8436
Total time (ms)	8005	2455	8886

Table 1: Inference time

It is possible to draw some conclusions from the results. First, the ontology load time does not depend on the number of instances of the Result class. This time value is due to the total number of instances and mainly to the number of SWRL rules. If the number of SWRL increases, the ontology load time will be higher. Also, the total time depends mainly on the computing time, which is really high. This time is the time spent by the AIRS to extract and process the results from the inference engine, and to read and write from the ontology. The inference time can be acceptable.

6. Conclusions

This paper proposes the application of ontologies, formal behaviour languages and semantic engines, to automatically choose the optimum response against a detected intrusion in a network

with heterogeneous detection sources that use different intrusion formats and syntaxes. The intrusion response ontology formally models all the elements used in the reaction process, and the inference rules are used by the semantic engine to infer new knowledge. Inference rules are specified using SWRL, a formal behaviour language.

The proposed approach has been tested in a specific scenario where an organization network is protected by the ontology-based AIRS, and a denial of service attack is detected. The results of these tests show the feasibility of using Semantic Web technologies to automatically choose the optimum response, due to its great expressiveness and flexibility. The main drawback is the computing time. Another drawback is the limited expressivity of SWRL, not allowing some logic operators, such as OR or NOT. This limitation results in an increased number of SWRL rules.

As a future work it should be interesting to use concurrent programming in order to reduce the computing time. Also, it could be better to use DL-safe rules instead of SWRL rules, because the SWRL is undecidable. However, the usage of Pellet solves this problem because Pellet interprets SWRL using the DL-Safe rules notion, which means rules will be applied to only named individuals in the ontology, ie, Pellet automatically converts SWRL rules to DL-Safe rules.

References

ABDOLI, F. and KAHANI, M. (2009): Ontology-based Distributed Intrusion Detection System. *Proceedings of the 14th International CSI Computer Conference (CSICC'09)*.

AZEVEDO, R.B., ROMMEL, E., DANTAS, G., FREITAS, F., RODRIGUES, C., SIQUEIRA, M.J., ALMEIDA, C., VERAS, W.C. and SANTOS, R. (2010): An Autonomic Ontology-based Multiagent System for Intrusion Detection in Computing Environments. *International Journal for Infonomics (IJI)*, 3(1): 182–189.

CARVER, CA. (2001): Adaptive Agent-based Intrusion Response. *PhD thesis*, Texas A&M University.

DEBAR, H., CURRY, D. and FEINSTEIN, B. (2007): RFC 4765: The intrusion detection message exchange format (IDMEF).

GÓMEZ-PÉREZ, A., FERNÁNDEZ LÓPEZ, M. and CORCHO, O. (2004): Ontological engineering.

HERZOG, A., SHAHMEHRI, N. and DUMA, C. (2007): An Ontology of Information Security. *International Journal of Information Security* 1(4): 1–23, 2007. Springer Verlag; 2004.

GUARINO, N (1998): Formal Ontologies and Information Systems. In *Proceedings of Formal Ontology in Information Systems, FOIS'98*; Trento, Italy, 6–8 June.

HORROCKS, I., PATEL-SCHNEIDER, P.F., BOLEY, H., TABET, S., GROSOF, B. and DEAN, M. (2004): SWRL: A semantic web rule language combining OWL and RuleML.

LI, W. and TIAN, S. (2008): XSWRL, an Extended Semantic Web Rule Language. *Second International Symposium on Intelligent Information Technology Application*.

LOPEZ DE VERGARA, J.E., VÁZQUEZ, E., MARTIN, A., DUBUS, S., and LEPAREUX, M-H. (2009): Use of Ontologies for the Definition of Alerts and Policies in a Network Security Platform. *Journal of Networks*, 4(8).

LOPEZ DE VERGARA, J.E., VILLAGRÁ, V.A. and BERROCAL, J. (2010): Benefits of Using Ontologies in the Management of High Speed Networks. *Proceedings of 7th IEEE International Conference on High Speed Networks and Multimedia Communications (HSNMC'04)*. Toulouse, France, 30 June–2 July 2004. *Lecture Notes in Computer Science*, 3079: 1007–1018, Springer-Verlag 2004.

MATEOS, V., VILLAGRÁ, V.A. and ROMERO, F. (2010): Ontologies-based automated intrusion response system. In *Proceedings of the 3rd International Conference on Computational Intelligence in Security for Information Systems (CISIS '10)*, 11–12 November.

MATEOS, V., VILLAGRÁ, V.A., ROMERO, F. and BERROCAL, J. (2012): Definition of response metrics for an ontology-based Automated Intrusion Response Systems. *Computers & Electrical Engineering*, 38(5): 1102–1114.

NOY, N.F. and MUSEN, M.A. (1999): An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support. In *Proceedings of the Workshop on Ontology Management, Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, Orlando, Florida, July.

PAPADAKI, M. and FURNELL, S.M. (2006): Achieving automated intrusion response: A prototype implementation. *Information Management & Computer Security*, 14(3): 235–51.

SMITH, K., WELTY, C. and McGUINNESS, D.L. (2004): OWL web ontology language guide. W3C recommendation; 10 February.
SOUAG, A., SALINESI, C. and COMYN-WATTIAU, I. (2012): Ontologies for Security Requirements: A Literature Survey and Classification. *CAiSE 2012 Workshops, LNBIP* 112: 61–69.
STAKHANOVA, N., BASU, S. and WONG, J. (2007-1): A cost-sensitive model for preemptive intrusion response systems. In *Proceedings of the 21st International Conference on Advanced Networking and Applications. AINA' 07*. IEEE Computer Society, Washington, DC, USA, 428–35.
STAKHANOVA, N., BASU, S. and WONG, J. (2007-2): A taxonomy of intrusion response system. *International Journal of Information and Computer Security*; 1(1/2): 169–84.
TSOUMAS, B. and GRITZALIS, D. (2006): Towards an ontology-based security management. In *AINA*, 985–992.
UNDERCOFFER, J., JOSHI, A. and PINKSTON, J. (2003): Modeling Computer Attacks: An Ontology for Intrusion Detection. In VIGNA, G., KRÜGEL, C. and JONSSON, E. (eds.) *RAID 2003. LNCS*, 2820: 113–135. Springer, Heidelberg.
VOROBIEV, A. and HAN, J. (2006): Security Attack Ontology for Web Services. In *SKG 2006*, 42. IEEE Computer Society.
WU, Y-S., FOO, B., MAO, Y-C., BAGCHI, S. and SPAFFORD, E. (2007): Automated adaptive intrusion containment in systems of interacting services. *Computer Networks*, 51(5): 1334–60.

Biographical Notes

Véronica Mateos

Verónica Mateos *is a researcher in the Department of Telematics Systems Engineering at the Technical University of Madrid (UPM). She received her PhD degree in telecommunication engineering from that university in 2013. Her research interests are in the area of security networks, and ontologies and semantic web. She also has participated in several national and international research projects.*

Victor A. Villagrá

Dr Víctor A. Villagrá *has been an associate professor in telematics engineering at the Technical University of Madrid (UPM) since 1992. He has been involved in international research projects related to network management, advanced services design and network security, as well as different national projects. He is the author or co-author of more than 70 scientific papers and is the author of a textbook about security in telecommunication networks.*

Julio Berrocal

Dr Julio Berrocal *is a full professor at Technical University of Madrid (UPM). He received a telecommunication engineering degree in 1983 and a PhD in telecommunications in 1986, both from UPM. He has been involved in several projects of the European Union R&D Programmes in management of telecommunication networks and services, multimedia broadband networks and emergency management. Professor Berrocal has authored or co-authored three books and more than 80 technical papers and reports, on topics such as communication protocol design and implementation, modelling of network management information and cybersecurity.*