A response toolkit to provide an active response against intrusions using Ontology-Based IRS

Danny S. Guamán[1], Julio C. Caiza[1], Verónica Mateos[2]

[1] Department of Electronics, Telecommunications and Information Networks, Escuela Politécnica Nacional, Ladrón de Guevara, Quito, Ecuador, E11-253.

[2] Department of Telematic Systems Engineering, Technical University of Madrid, Ciudad Universitaria 30, Madrid, Spain.

Autor para correspondencia: danny.guaman@epn.edu.ec

Fecha de recepción: 24 de octubre de 2014 - Fecha de aceptación: 28 de octubre de 2014

ABSTRACT

Active response systems are intended to run an automatic response against an intrusion. However, running an automatic response is not a trivial task because the execution cost could cause a greater negative effect than the intrusion itself. Also, the system should have a broad set of responses and an algorithm to select the optimal response. This paper proposes a response toolkit that is integrated into an ontology-based IRS to allow automatic execution of the best response against a detected intrusion. A set of host-based and network-based responses that can be performed by an IRS is presented. The response execution is performed by several plugin-based agents that have been distributed over the network. The verification of this proposal is made in a defacement attack case with satisfactory results.

Keywords: Network security, intrusion response, active response.

RESUMEN

Los sistemas de respuesta activa tienen por objetivo ejecutar una respuesta en contra de una intrusión de forma automática. Sin embargo, ejecutar una respuesta automáticamente no es una tarea trivial ya que el costo de ejecutar una respuesta podría ser más grande que el efecto que cause la intrusión propiamente dicha. También, el sistema debe contar con un amplio conjunto de acciones de respuesta y un algoritmo que seleccione la respuesta óptima. Este artículo propone un toolkit de respuestas que será integrado a un IRS basado en Ontologías para permitir la ejecución automática de la mejor respuesta cuando una intrusión es detectada. Se presenta un conjunto de respuestas basadas en host y basadas en red que pueden ser ejecutadas por el IRS, dicha ejecución es llevada a cabo mediante agentes basados en plugins que han sido distribuidos en la red. Finalmente, se realiza la verificación del sistema propuesto, tomando como caso de uso un ataque de defacement obteniéndose resultados satisfactorios.

Palabras clave: Seguridad de redes, respuesta a intrusiones, respuestas activa.

1. INTRODUCTION

Network security has required constant research to evaluate and improve the access control mechanism in order to mitigate attacks, which are performed for different purposes. Solutions such as firewalls, to implement a security policy, or routers to define access control lists cannot guarantee full protection against existing attacks or new attacks (Ingham & Forrest, 2002). Then, the intrusion detection systems emerge as a new defense mechanism to detect a wider range of attacks that violate the integrity, confidentiality and availability of a resource.

Intrusion Detection Systems (IDS) have been subject of a constant review and evolution, enabling the emergence of Intrusion Prevention Systems (IPS) and Intrusion Response Systems (IRS) (Anuar *et al.*, 2010). These three systems: IDS, IPS, and IRS are intended to monitor and detect anomalous behavior in a computer system or in the network activity; however, they differ in the way how they react to an incident:

- An IDS is able to detect an intrusion and generate a warning or alert to the system administrator, who performs a response. The problem in this approach is the long time required until the administrator performs a response. In this time the attacker could have achieved his goal (Stakhanova *et al.*, 2007).
- An IPS performs the same detection process than the IDS, but it is not limited to sending a simple alert. This system is able to execute a proactive action to prevent an attack. The response usually involves a firewall or an access control device that must necessarily be located in line with the malicious traffic to block it (Rowland, 2002).
- An IRS also detects an intrusion and executes a response, but differs from the IPS because this response is not necessarily executed on devices in line with malicious traffic. The IRS response is reactive because its main objective is to reduce the damage caused by an intrusion. The IRS should have a catalog with several response actions and should provide a mechanism to evaluate the cost caused by the intrusion vs. the cost of executing a response. The evaluation allows selecting the best response (Stakhanova *et al.*, 2007).

There are two research areas: intrusion detection techniques (Tavallaee *et al.*, 2010; Nazer & Selvakumar, 2011; Mallissery *et al.*, 2011), and intrusion response techniques (Stakhanova *et al.*, 2007; Anuar *et al.*, 2010; Shameli-Sendi *et al.*, 2012). This work is focused in the second one, the intrusion response systems.

In previous work, Mateos *et al.* (2010) proposes an Ontology-Based Intrusion Response System. This system has the ability to adapt its decision to select a response action depending on the context and the cost of running it. This paper is intended to contribute to the ontology-based IRS through a review, proposal and implementation of a response toolkit that will be adapted to the response executor module. This contribution will help to validate the entire system.

The rest of the paper is organized as follows. Related work is done in Section 2. A review of the Ontology-based IRS architecture is carried out in Section 3. Section 4 presents a review, proposal and implementation of a response toolkit that has been integrated to Ontology-based IRS. In Section 5, the verification of the entire system is done through a use case. Finally, Section 6 shows the conclusions and possible future work in this area.

2. RELATED WORK

The active response against intrusion is not new. In recent years, several proposals related with intrusion response systems have been done. Thames *et al.* (2008) describes a distributed firewall and active response architecture to provide preemptive protection. The authors designed an architecture based on the concept of hosts within a trusted domain of administration. After it detects an intrusion creates blocking policies on firewall against the anomalous host. These blocking policies are shared with the neighborhood members of the domain of administration. Another work presents a system which combines heterogeneous intrusion detection systems with distributed firewall system (Han *et al.*, 2006). This system detects and prevents intrusion originated from intranet or Internet.

Also, several works have contributed with proposals of systems that trigger a dynamic response against a detected intrusion. These systems infer the suitable response and trigger it automatically. Stakhanova *et al.* (2007) and Shameli-Sendi *et al.* (2012) have gathered the main contributions of these systems. They classify the IRS by ability to adjust the response, by response selection mechanism, by time of response and, finally, by response cost model.

The use of ontologies in network security has been related to how to classify and model security information. Souag *et al.* (2012) review, analyze and classify security ontologies. Undercoffer *et al.* (2003) present specific security ontology for intrusion detection postulating a model of computer attack. This ontology models concept related to intrusion, such as: attack, system, component, host and consequence. Abdoli & Kahani (2009) present attack ontology for intrusion detection in distributed system.

None of the mentioned works completely models the intrusion response process; however, in previous work Mateos & Vilagrán (2013) reuse some of them and propose an Ontology-Based Intrusion Response System. This system has the ability to adapt its decision to select a response action depending on the context and the cost of running it. The proposed architecture uses formal languages to specify behavior. It helps to understand the semantics associated with intrusion alerts generated by different IDS and ensure consistency semantics in a heterogeneous environment.

Also, we can found that most proposed architectures to provide active response only interact with firewalls. However, it could be necessary interact with other security components to execute a complete response. For this reason, this work is intended to contribute to ontology-based IRS through a review, proposal and implementation of a response toolkit which interact with other security components. Host-based and network-based responses can be executed for the based-ontology Intrusion Response System through the architecture which was proposed in a previous work (Guamán & Mateos, 2014).

3. ONTOLOGY-BASED INTRUSION RESPONSE SYSTEM

The Ontology-based IRS is an automatic intrusion response system, whose main contribution is the addition of semantic coherence to the set of IRS' feature. That is essential because in a heterogeneous network environment there are several types of IDS and each handles alerts in different formats (Mateos *et al.*, 2010). The IRS architecture (Fig. 1) selects the optimal response from a set of recommended ones. This is achieved by using an ontology that is represented by a formal language. Classes included in the ontology are used to represent certain policies that are defined by the administrator; these policies are used for semantic reasoning to infer the best response. Below is a brief description of the architecture of the different modules.

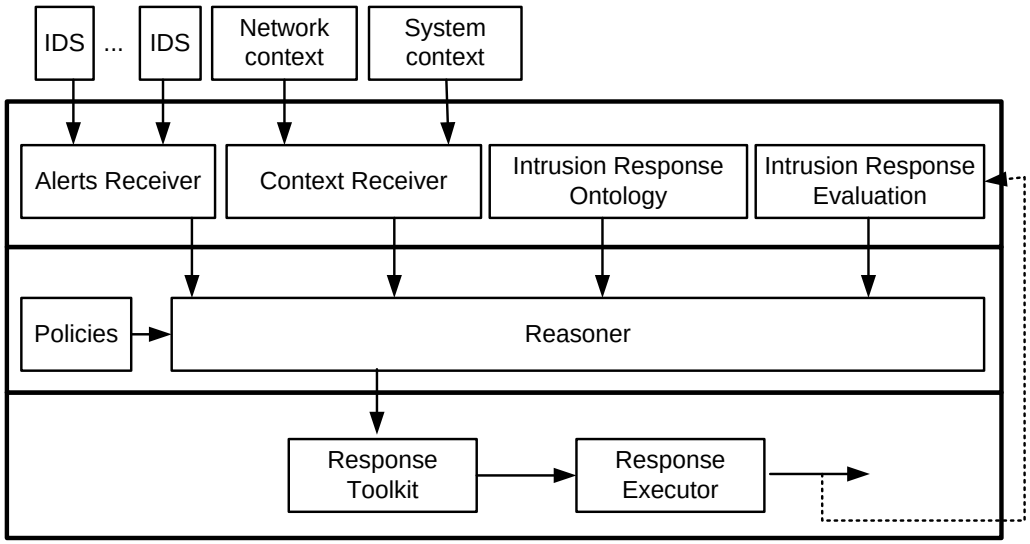

Figure 1. Ontology-based intrusion response system architecture (Mateos *et al.*, 2010).

Alerts Receiver: It receives intrusion alerts from different IDSs, with different formats and syntax, and maps the included fields to their equivalent concepts defined in the ontology.

- Network Context: It calculates a parameter called network anomaly based on the difference between two network traffic snapshots. The first one is obtained under normal operating conditions, and the second one is obtained when an intrusion is detected.
- System Context: It calculates a parameter called system anomaly based on the difference between two states, that evaluate the number of active processes, number of zombie processes, memory used and disk space, before and after the attack.
- Context Receiver: It receives the network anomaly and system anomaly parameters and then maps to their corresponding concepts defined in the ontology.
- Intrusion Response Ontology: It defines the intrusion response domain, which is necessary to infer the optimal response. The ontology, shown in Fig. 2, models the problem domain identifying the following entities: network, system components, intrusion detection systems, intrusion response systems, intrusion alerts, network context, system context, responses and results. Each of these entities is represented by classes, properties and relationships, using the standard language OWL (Ontology Web Language) and Protégé tool.
- Policies: They are composed of a rule set that define the IRS behavior. These rules, called response metrics, are defined by the system administrator using the SWRL language (Semantic Web Rule Languages). Response metrics and algorithm for selection of the optimal response is presented by Mateos *et al.* (2012).
- Reasoner: Semantic reasoner is a piece of software able to infer logical consequences from a set of facts or axioms succeeded. Axioms are written by the OWL and SWRL languages. It is the main component of IRS because it is responsible of inferring the optimal response to a given intrusion. The reasoner takes as input the policies and an ontology instance. Considering the Ontology-based IRS requirements, Pellet reasoner (Sirin *et al.*, 2007) was chosen.

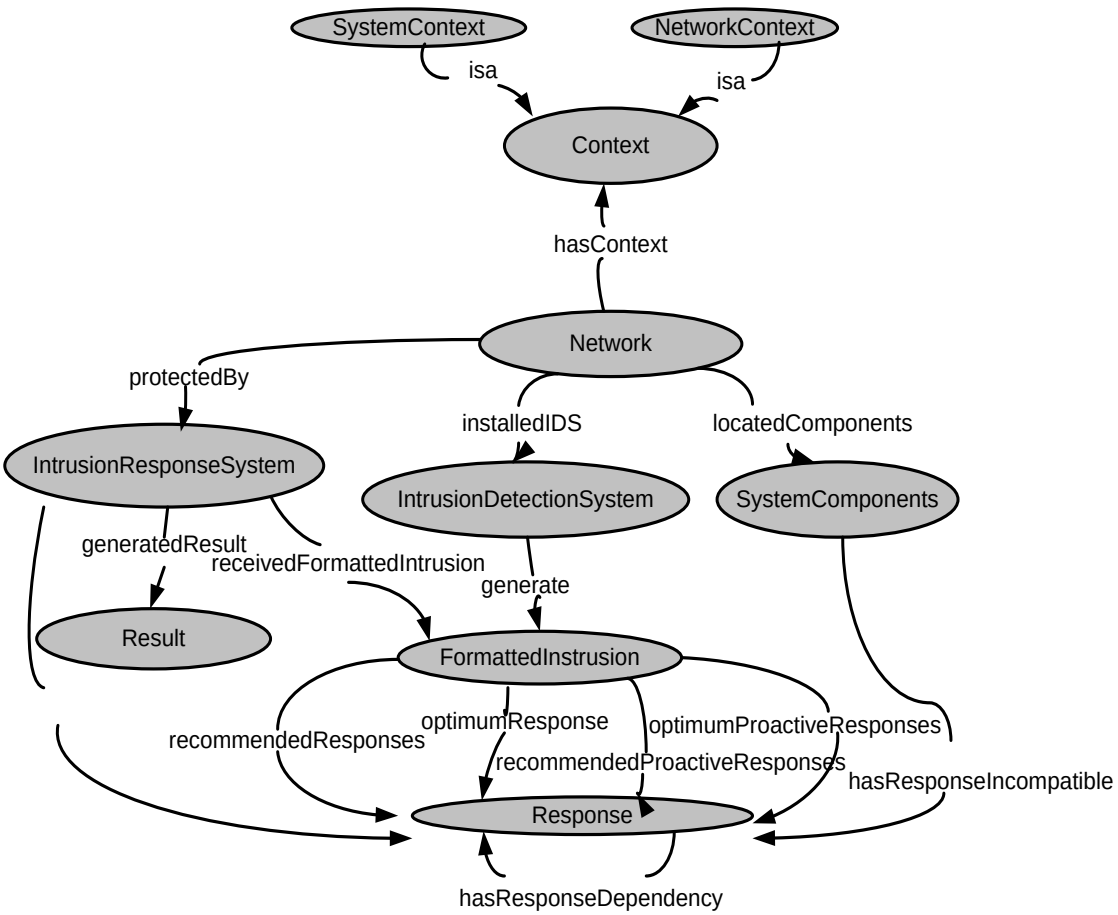

Figure 2. Intrusion response system ontology.

- Response Toolkit: It is the set of response actions that is available to the reasoner. Once the reasoner infers the optimal response, one or more actions could be selected and executed by the response executor module. It is discussed in the next section.
- Response Executor: It interacts with the reasoner through the response toolkit receiving the necessary parameters to execute an action over security components, such as firewalls, routers and hosts. Plugin-based response executor architecture is presented in a previous work of the authors (Guamán & Mateos, 2014).
- Intrusion Response Evaluation: It evaluates if a previous response was satisfactory. The result is stored in a historical database and is used for future responses.

4. RESPONSE TOOLKIT

4.1. Response actions

After the IRS detects an intrusion it sends an alert to the reasoner which based on to their responsiveness performs active or passive response (Villagrán, 2009).

- Passive responses: Their purpose is to record the intrusion but not executes any action to prevent or mitigate the damage caused by an attack. In this case, the administrator manually performs a response. Some examples of passive responses include sending an email or text message, sending an SNMP trap to a management console and registry logs in a file or incident database.
- Active responses: It attempts to automatically mitigate the attack through the execution of an action without requiring the administrator intervention. We have classified the active responses into four categories:
  - *Protection responses*: They try to cancel any type of interaction between the attacker and the victim machine. Examples of such responses include: adding a policy on a firewall to block connection attempts, adding rules to the access control lists on a router, killing the processes associated with connections to a specific IP address, blocking a port; killing a service or disabling a user.
  - *Recovery responses*: They try to restore an attacked resource to its previous state. Examples of these responses are: restoring files after a website defacement attack, restoring a BDD after unauthorized changes, restoring of an attacked host to its previous state, etc.
  - *Deception responses*: They are intended to simulate vulnerable system in order to attract attackers. Then, the administrator could collect information about the attack context, for example: who performed the attack?, the attack target?, and techniques employed? Honeypots and honeynets are some technologies used to deploy these responses.
  - *Reaction responses*: They are intended to execute a counter attack. Their purpose may be to discover the attacker identity or find personal information; however, laws should be considered because an attack can be performed from a different jurisdiction than the victim. The automatic execution of any active responses is not a trivial task and should consider two things. Firstly, it is necessary to evaluate the cost of implementing an active response because it could be higher than the intrusion itself. In our architecture the IRS reasoner is responsible to evaluate it. Secondly, the definition of response actions depends on the organization in which these will be implemented. A risk analysis provides a good approximation to protect the most important resources and give us a clear sight about what response actions should be implemented (Broder & Tucker, 2011).

A set of response action that can be executed by IRS is presented in Table 1. We have classified in: network-based responses, which have the ability to interact directly with the network traffic, and; host-based responses, which have the ability to interact with processes, services, connections and users hosted on a specific operating system.

Table 1. Host-based and network-based responses.

Response action description	Pv	P	D	R	C
To send a notification to the administrator.	√				
To enable additional analysis tools.	√			√	
To backup manipulated files.	√				
To trace the attacker connection to gather information.	√				√
Host-based					
To deny selective or full access to a file.		√			
To delete a modified file.		√			
To allow manipulation of a fake file.			√		
To restore a manipulated file with a backup file.				√	
To restrict user activity.		√			
Disable a user account.		√			
Shut down the compromised host.		√			
Shut down the compromised service.		√			
To reload suspicious process.		√		√	
To block suspicious system calls.		√			
To delay suspicious system calls.		√			
To kill suspicious process.		√			
Network-based					
Add or delete filtering rules on a firewall.	√	√			
To reload target system.		√		√	
To block suspicious incoming and outgoing network connections	√	√			
To block ports or IP addresses.	√	√			
To deploy a honeypot or honeynet.	√		√		
Spoofing TCP RST or ICMP Message Unreachable to terminate TCP and UDP connections.		√			

Legend: (Pv) Passive response (P) Protection active response; (D) Deception active response; (R) Recovery active response; (C) (Reaction active response).

4.2. *Implementation and integration of the response toolkit to ontology-based IRS*

In previous work response executor architecture was proposed. This was integrated into ontology-based IRS. Now, we present a data model that facilitates the registration of new response actions into the toolkit. We have also implemented several proof of concept of response actions (plugins) to validate the entire system. The Fig. 3 shows the components over a communications network (A) and modular decomposition (B) of response executor.

- IDS: They are external systems, which after detecting an intrusion will send alerts to reasoner. Open source IDS were used: the host-based IDS called OSSEC; and network-based IDS called Snort.
- Reasoner: It receives an intrusion alert and infers the optimal response from the available actions in the response toolkit. The reasoner invokes the Central Execution Module sending the necessary parameters to execute a response action; these parameters are obtained from intrusion alerts or by using network management tools like Nagios and Net-SNMP.
- Response toolkit: We have defined a database which contains the response actions that can be executed on a security component. The data model defined is shown in Fig. 4. Database information must be provided by the administrator:
  - *Group Executor Agents*: This consists by the agents to whom response request is sent. Each execution agent contains the following information for its location: IP address, port and password encryption.
  - *SIDS Group*: It is comprised of a Signature Identifiers (SID) set for which a specific response is executed.

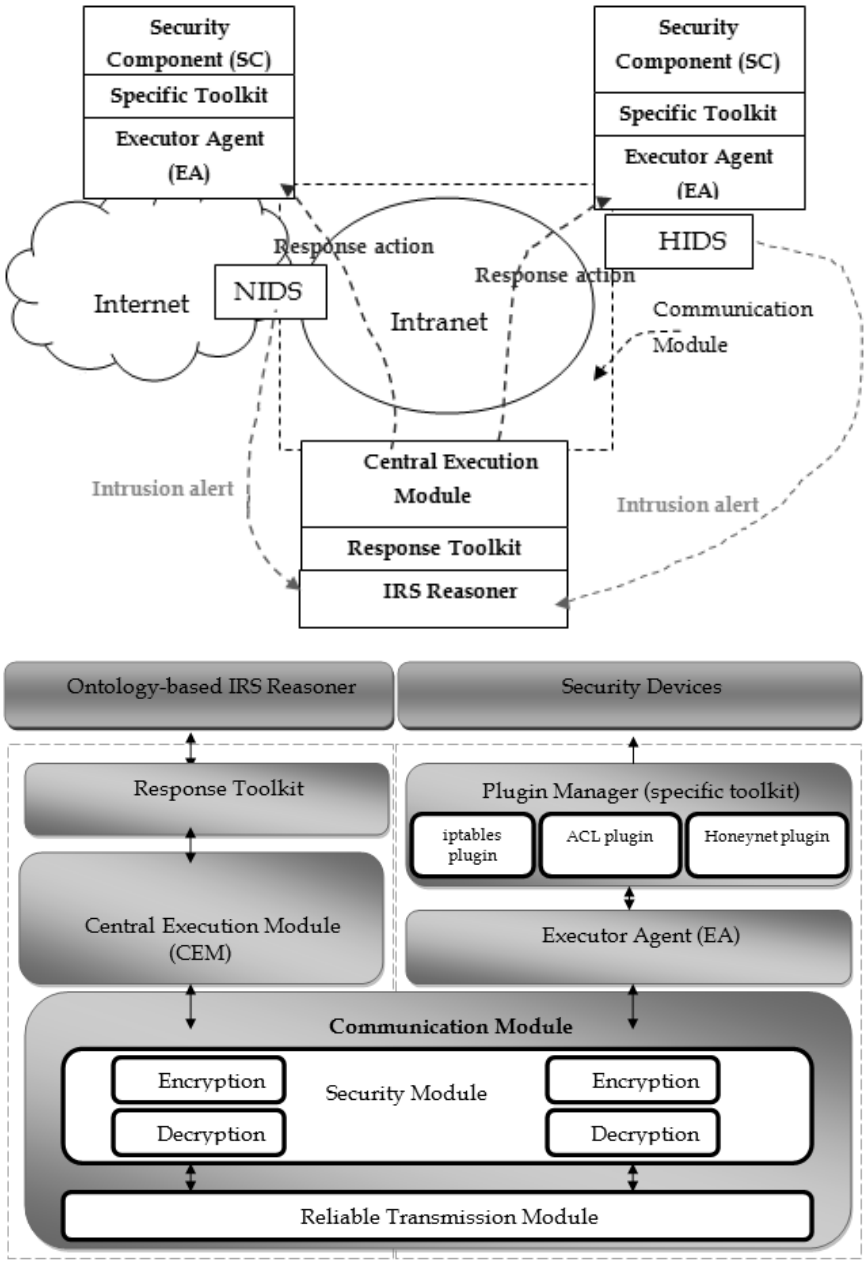

Figure 3. Response executor module: (Top) Distribution of components over communication network. (Bottom) Modular descomposition.

- *Plugin*: It indicates which executor agent plugin will perform a response action.
- *Response Action*: It defines the duration, mode, action, plugin, signature identifier and executor agents to execute a response.
- *Response Action Parameters*: They are the parameters provided by the reasoner. The IP addresses and ports of the attacker and the victim are examples of them.

Table 2 (see further) shows a description of some response actions that have been implemented as a proof of concept and have been integrated into the Response Toolkit.

- Central Execution Module (CEM): It is responsible for two tasks: it constructs a response request and it identifies the executor agents which will sent the response request. To build a response request the CEM receives parameters from the reasoner and from the response toolkit. Finally, it invokes the communication module.

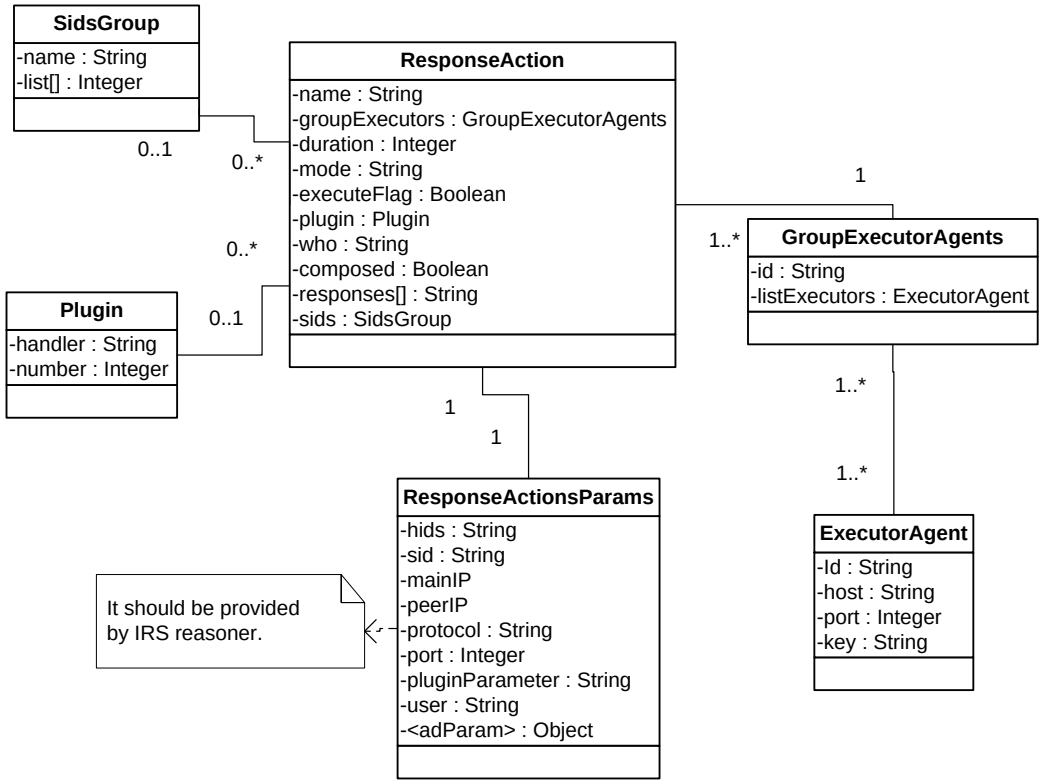

Figure 4. Response toolkit data model.

- Communication module (CM): This module provides of a common framework for communication between the CEM and executor agents giving reliable and safe delivery services.
- Executor Agents: This is a background service that waits for a response request. When a request arrives it selects the appropriate plugin and executes several commands on a security component.
- Plugin Manager (Specific Toolkit): This solves the problem related with the security components heterogeneity. We propose encapsulating control logic of a response within a plugin. Thus, the response executor can be used as a scalable platform to implement new response actions on several security components without the need of modifying other modules. Therefore, a query interface and a registration interface have been defined. The query interface allows to executor agent to identify the plugin that has been registered. Meanwhile, the registration interface lets adding a new plugin providing routines for: initialization, parsing, execution, finalization and keepalive. In practice a new plugin is recorded in a header file called plugin.h, while specific plugin parameters are recorded in a file called agent.conf.

5. USE CASE

This section shows the Ontology-based IRS operation during a web server defacement attack. To perform the use case an ad hoc network was deployed (Fig. 5). Each network component is shown in Table 3. In order to show the results and the interaction between modules implemented, the process is divided into three activities: detection, inference and execution.

Table 2. Response actions (proof of concept) added to response toolkit.

Plugin Name	Parameters provided by IRS Reasoner	Parameters defined in agent.conf file	Action response description
ciscoacl	• Attacker IP address • Victim IP address • Victim port • Transport protocol used	• Router IP addres • Telnet password • Enable password • Initial ACL file • ACL name (optional) • Physical interface to apply ACL • ACL type: in/out • TFTP Server IP address	This plugin adds an access control list to a Cisco router. The same plugin can be used to add an ACL to filter: input, output, input and output, or specific connection traffic, associated with the attacker IP address. This plugin can operate in two ways: 1) Sending individual commands to the router via a remote connection, and 2) Sending a file modified running-config from a TFTP server.
ipt-block	• Attacker IP address • Victim IP address • Victim port • Transport protocol used	• Physical interface	This plugin adds a rule to Filter Table in an iptables firewall. The same plugin can be used to add a rule that filters: input, output, or input and output traffic related with the attacker IP address.
vnxhn		• Path of VNX configuration file.	This plugin displays a honeynet using a virtualization tool called Virtual Network over Linux (VNX) (Galán *et al.*, 2009). The VNX configuration file creates a virtual network where there are tools to analyze the attacker's modus operandi.
iptredirect	• Attacker IP address • Victim IP address • Victim port • Transport protocol used	• IP Address to where the traffic is redirected. • Port to which the traffic is redirected.	This plugin redirects: input, output, input and output traffic, to an IP address and port defined as a parameters. It is used together with the vnxhn plugin.
cisconullroute	• Attacker IP address • Victim IP address • Victim port • Transport protocol used	• Router IP address • Password enable • Password telnet	This plugin adds a null route to a Cisco router. The same plugin can be used to add an ACL to filter: input, output, input and output, or specific connection traffic, associated with the attacker IP address.
upload-backup-ftp download-backup-ftp		• FTP IP address • FTP username • FTP password • Path of source file • Path of target file • Path of source directory	This plugin is intended to backup/restore a user defined file to/from an FTP server. If a file with the same name exists save a copy as a backup.
backup-mysql restore-mysql		• MySQL Server IP address • BDD username • BDD password • BDD name • Backup file name	This plugin is intended to backup/restore a MySQL database.
disable-user	• Linux username		This plugin is intended to disable a user over a Linux operating system.
close-port	• Victim Port • Transport protocol used		This plugin closes a port on the attacked host.

Table 2 (continued). Response actions (proof of concept) added to response toolkit.

Plugin Name	Parameters provided by IRS Reasoner	Parameters defined in agent.conf file	Action response description
host-deny	• Attacker IP address	• Path of host.deny file	This plugin adds a rule in the host.deny file to deny access to the attacked host.
close-connections	• Attacker IP address • Victim IP address • Victim port • Transport protocol used		This plugin closes all host established connections that matches with the attacker IP, victim IP, attacker and victim IP or specific connection.
Email	• Attacker IP address • Victim IP address • Victim port • Transport protocol used	• SMTP IP address • SMTP port • Source email • Target email	This plugin ends an email with information about the intrusion.

Table 3. Test network components.

Component	Features
RFW y RINT	Cisco Router C3640 Version 12.3(26)
ATT (Atacante)	GNU/Linux Ubuntu 10.04.3 LTS with BackTrack 5
INT1-1, INT1-3, INT2-1 e INT2-5	GNU/Linux Ubuntu 11.04
INT1-2, INT1-4, INT2-2, INT2-3, INT2-4 e INT2-6	Windows XP Service Pack 3
INT3-1 e INT3-2	GNU/Linux Ubuntu 11.04 HIDS OSSEC version 2.7
DMZ-1	GNU/Linux Ubuntu 8.0.4 with Metasploitable
DMZ-2	GNU/Linux Ubuntu 11.04
IDS-1, IDS-2 e IDS-3	GNU/Linux Ubuntu 10.04.2 TTS NIDS Snort version 2.9.2.3
Ontology-based IRS	GNU/Linux Ubuntu 12.04.1 LTS PC: Dell XPS L502X, 8GB RAM, Processor: 2.6GHz.

5.1. Detection

- E1: the attacker (10.1.200.22) attempts a defacement attack to a web server. The attack result is the modification of index.html file, which is located in the /var/www/ web server directory (192.168.100.130).
- E2: The OSSEC HIDS syscheckd installed in the DMZ-2 server detects that index.html file has been modified and sends an alert in syslog format to the IRS reasoner. The reasoner receives the alert through port 512.

5.2. Inference

In this case the reasoner infers a composite active response, whose name is relayAttack. This composite response is intended to redirect the attacker to a honeynet to analyze their modus operandi, to restore the modified file and notify the administrator through email. The relayAttack response consists of four simple actions performed by four different plugins:

- *honeynetDeployment*: It runs on the executor agent through hnvnx plugin to deploy the honeynet.

redirectTraffic: It runs through the ipt-redirect plugin on the executor agent to interact with a perimeter firewall and redirects all traffic to the honeynet deployed.

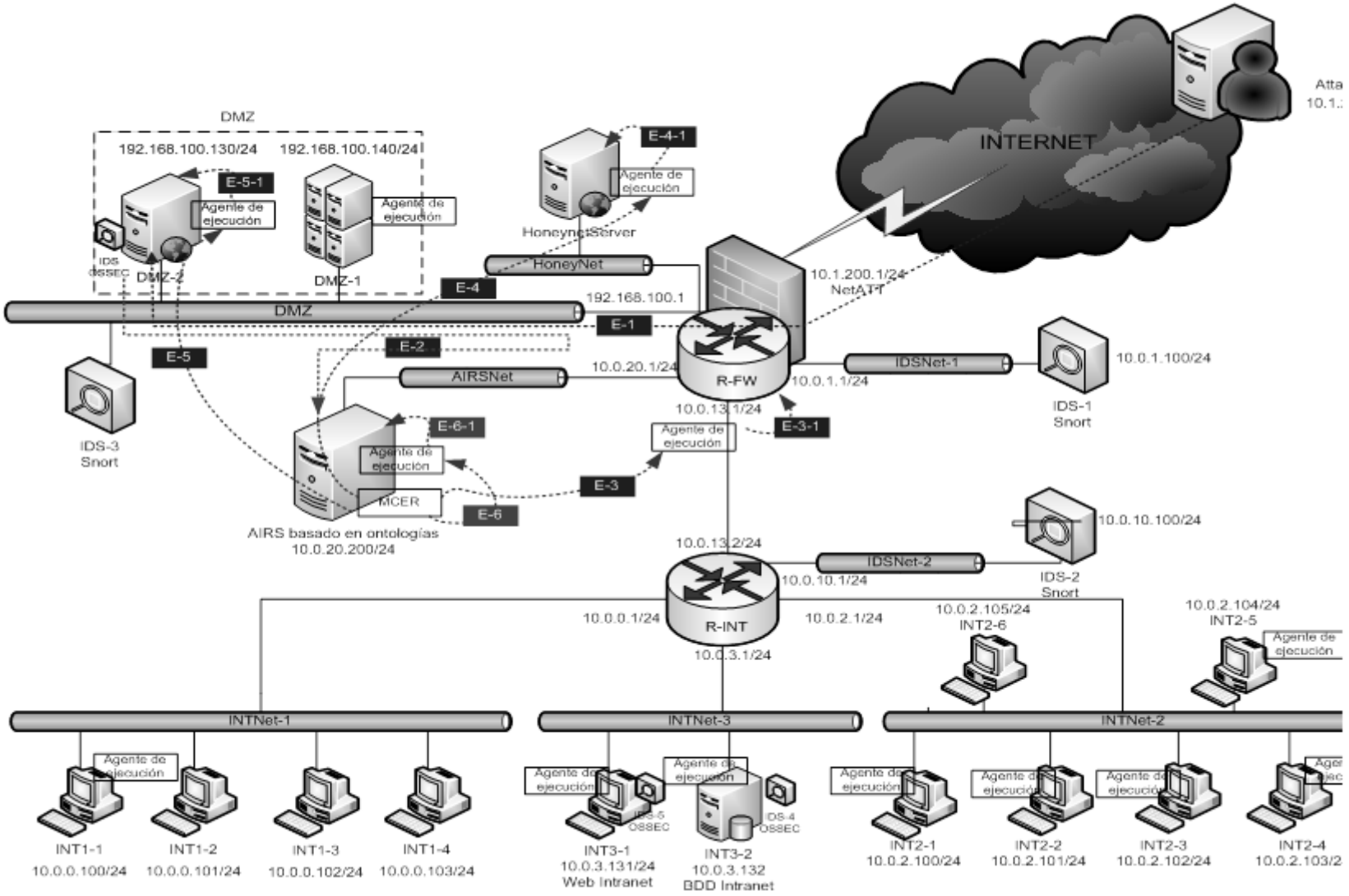

Figure 5. Defacement attack in the test network.

- *restoreIndex*: It uses the download-backup-ftp plugin to replace modified index.html file from an FTP server.
- *mailNotification*: It sends an e-mail notification to the administrator by using the email plugin.

The Central Execution Module establishes four connections through the Communication Module (one connection per plugin) to send response requests to each executor agent.

5.3. Execution

- E3 and E3-1: The executor agent through the ipt-redirect plugin adds two rules to iptables perimeter firewall. The first adds a rule to the FORWARD chain of FILTER table to discard all traffic from the attacker (10.1.200.22). The second adds a rule to the PREROUTING chain of NAT table to redirect traffic from the attacker (10.1.200.22) to honeynet virtualized host (192.168.100.150).
- E4 and E4-1: The executor agent executes the second action through hnvnx plugin. The honeynet is deployed using the Virtual Network over Linux (VNX). Before the implementation of this action, the administrator must create the VNX file with the virtual environment.
- E5 and E5-1: The executor agent located in the DMZ-2 server (192.168.100.130) through download-backup-ftp plugin restores the index.html file from a secure FTP server. This plugins performs three tasks: first, it establishes a connection and authenticates the FTP server; second, it makes a backup of the modified file for later analysis, and; finally, it downloads the index.html file from the FTP server and replaces the modified file.
- E6 and E6-1: The executor agent, through email plugin, sends an email to the administrator indicating that a defacement attack was detected. It establishes a connection to the SMTP server which acts as a relay to send the email.

Finally, it was corroborated the successful result about composite response execution carried out by the Ontology-based IRS:

- *Recovery Response*: The modified index.html file was restored to its original state.
- *Deception Response*: A honeynet was deployed using the VNX tool and the attacker traffic was redirected using iptables firewall.
- *Passive Response*: An email was sent to the network administrator.

6. CONCLUSIONS

The automatic response executed by the Ontology-based IRS can reduce the time window to react against an intrusion. In addition, unlike other tools such as SnortSam, Fwsnort or Suricata, the IRS does not always perform the same response against a detected intrusion; instead, the reasoner selects the optimal response.

This paper presents a review of passive and active responses that can be executed by an IRS. We present a data model for the response toolkit and implemented several response actions as a proof of concept. The response control logic is encapsulated within plugins; in addition, these plugins run on executor agents that have been distributed over the network. The use case demonstrates the benefits of this work. We observed that the IRS can execute complex responses that have been built based on simple actions (plugins). Also, our system can interact simultaneously with several safety components.

Finally, we can mention that the use of multiple distributed IDS (in our case HIDS and NIDS) allows detecting a wider range of intrusions. This allows detecting attacks carried out from within or outside the organization network.

REFERENCES

Abdoli, F., M. Kahani, 2009. Ontology-based distributed intrusion detection system. In: *Computer Conference, 2009. CSICC 2009. 14th International CSI* (IEEE), 65-70.

Anuar, N.B., M. Papadaki, S. Furnell, N. Clarke, 2010. An investigation and survey of response options for Intrusion Response Systems (IRSs). In: *Information Security for South Africa (ISSA)*, 1-8.

Broder, J.F., G. Tucker, 2011. Risk analysis and the security survey (4th ed.). *Butterworth-Heinemann, Elsevier*, 348 pp.

Galán, F., D. Fernández, W. Fuertes, M. Gómez, J.E.L. de Vergara, 2009. Scenario-based virtual network infrastructure management in research and educational testbeds with VNUML. *Annals of Telecommunications-Annales Des Télécommunications*, 64(5-6), 305-323.

Guamán L.D.S., V. Mateos, 2014. Arquitectura Distribuida para la Respuesta Automática a Intrusiones en un IRS Basado en Ontologías. *Revista Politécnica*, 33(1).

Han, H., X.-L. Lu, L.-Y. Ren, B. Chen, 2006. Taichi: An open intrusion automatic response system based on plugin. In: *International Conference on Machine Learning and Cybernetics* (IEEE), 66-77.

Ingham, K., S. Forrest, 2002. A history and survey of network firewalls. *University of New Mexico, Tech. Rep*., 42 pp.

Mallissery, S., J. Prabhu, R. Ganiga, 2011. Survey on intrusion detection methods. In: *3rd International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2011)*, (IET), 224-228.

Mateos V., V. Villagrán, 2013. Application of ontologies and formal behavior definitions for automatic intrusion response systems. *University of Murcia / Universidad Politécnica de Madrid*, Spain, 9 pp.

Mateos, V., V. Villagrán, F. Romero, J. Berrocal, 2012. Definition of response metrics for an ontology-based automated intrusion response systems. *Comput. Elec. Eng.* 38(5), 1102-1114.

Mateos, V., V. Villagrán, F. Romero, 2010. Ontologies-based automated intrusion response system. In: *Computational Intelligence in Security for Information Systems.* Springer, 99-106.

Nazer, G.M., A.A.L. Selvakumar, 2011. Current intrusion detection techniques in information technology-a detailed analysis. *EJSR*, 65(4), 611-624.

Rowland, C.H., 2002. Intrusion detection system. Google Patents. Retrieved from *http://www.google.com/patents/US6405318*.

Shameli-Sendi, A., N. Ezzati-Jivan, M. Jabbarifar, M. Dagenais, 2012. Intrusion response systems: survey and taxonomy. *IJCSNS*, 12(1), 1-14.

Sirin, E., B. Parsia, B.C. Grau, A. Kalyanpur, Y. Katz, 2007. Pellet: A practical OWL-DL reasoner. *Software Engineering and the Semantic Web*, 5(2), 51-53.

Souag, A., C. Salinesi, I. Comyn-Wattiau, 2012. Ontologies for security requirements: A literature survey and classification. The *2nd International Workshop on Information Systems Security Engineering WISSE'12 in conjunction with the 24th International Conference on Advanced Information Systems Engineering (CAiSE'12)*, Gdansk, Poland, 8 pp.

Stakhanova, N., S. Basu, J. Wong, 2007. A cost-sensitive model for preemptive intrusion response systems. In: *21st International Conference on Advanced Information Networking and Applications* (AINA '07), 428-435.

Stakhanova, N., S. Basu, J. Wong, 2007. A taxonomy of intrusion response systems. *International Journal of Information and Computer Security*, 1(1), 169-184.

Tavallaee, M., N. Stakhanova, A.A. Ghorbani, 2010. Toward credible evaluation of anomaly-based intrusion-detection methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40(5), 516-524.

Thames, J.L., R. Abler, D. Keeling, 2008. A distributed firewall and active response architecture providing preemptive protection. In: *Proceedings of the 46th Annual Southeast Regional Conference on XX* (ACM), 220-225.

Undercoffer, J., A. Joshi, J. Pinkston, 2003. Modeling computer attacks: An ontology for intrusion detection. In: Vigna, G., E. Jonsson, C. Kruegel (Eds.). *Recent Advances in Intrusion Detection*. RAID 2003, LNCS 2820, Springer-Verlag Berlin Heidelberg, 113-135.

Villagrán, V., 2009. *Seguridad en Redes de Telecomunicación* (1st ed.). España: Fundación Rogelio Segovia para el Desarrollo de las Telecomunicaciones.