universidad de león

RIaSC

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

Evolving from a static toward a proactive and dynamic risk-based defense strategy

Pilar Holgado*, Manuel Gil Pérez†, Gregorio Martínez Pérez†, and Víctor A. Villagrá*

* Departamento de Ingeniería Telemática, Universidad Politécnica de Madrid, 28040 Madrid, Spain
† Departamento de Ingeniería de la Información y las Comunicaciones, University of Murcia, 30071 Murcia, Spain
Email: pilarholgado@dit.upm.es, mgilperez@um.es, gregorio@um.es, villagra@dit.upm.es

***Abstract*—The automatic prevention, detection, and reaction for intrusion management has been a key issue for years, focusing on the use of IDS-based approaches. In addition, dynamicity and the changing nature of the technology and threats have led to consider other approaches. In this paper, we present two R&D projects whose purposes are addressing the above shortcomings. First, we present the RECLAMO project, where an architecture for an Automated Intrusion Response System is proposed to divert a given attack to a honeynet, dynamically built based on the attack information. Secondly, we also describe an ongoing R&D project, called DHARMA, where an efficient Dynamic Risk Assessment and Management is proposed to measure the risk level on the organization's assets at real time, taking the required actions as a response from a proactive defense model.**

I. Introduction

Automatic prevention, detection and reaction systems for intrusion management is a key topic in the last few years [1]. Yet, most of the solutions have a narrow scope and certain difficulties and limitations when dealing with large scale and distributed attacks like coordinated spam, phishing attacks or DDoS [2]. To this end, concepts like autonomic system, trust and reputation management, collaborative intrusion detection and prevention systems, virtualized honeynets, and semantic web should be part of novel IDS/IPS systems.

Despite the progress in intrusion management, dynamicity and heterogeneity should consider other alternatives that take into account input from a large pool of heterogeneous sensors and contextual changes in the organization, beyond traditional network attacks. In this context, new dynamic risk assessment and management processes have emerged to provide an answer to this shortcoming, allowing the current solutions to acquire a dynamic assessment of the risk of the organizational assets and the continuous evolution of threats. A dynamic risk assessment and management allow updating the risk level on the assets of an organization at real time, as well as dealing with dynamicity and the changing nature of the technology and threats.

A. Our contributions

As a first contribution, we propose an approach to intrusion response, building and deploying honeynets where the attacks will be diverted. This is a result of the RECLAMO (*Virtual and Collaborative Honeynets based on Trust Management and Autonomous Systems applied to Intrusion Management*) project, where honeynets are created ad-hoc and optimized for each attack, in order to obtain as much information as possible from each [3], [4]. RECLAMO is aimed at designing and creating an advanced Automated Intrusion Response System (AIRS) to enhance the current attack detection and reaction proposals. Self-protection is the key concept driving the components of RECLAMO, providing a way of inferring the most appropriate response for a given intrusion, taking into account not just the intrusion, but also other related parameters, such as the context and the confidence on the network sources. This information is evaluated with a set of security metrics represented in a formally defined behavior specification language, in order to reason and to infer the most appropriate response.

As stated before, dynamicity and heterogeneity is key for making automated management of intrusions a reality. As a second contribution, we propose an efficient Dynamic Risk Assessment and Management (DRAM), which is part of a project called DHARMA (*Dynamic Heterogeneous threAts Risk Management and Assessment*) [5]. This is a multilevel architecture with a large number of heterogeneous sensors capturing changes in the organization context. The DHARMA framework enables to deploy specific sensors, integrating their information in a DRAM engine that will provide updated information on the risk levels to allow a quick reaction and minimizing the exposure time to potential risky situations for the organization. As a possible reaction of the DRAM is taking into account the AIRS proposed in RECLAMO.

B. Organization of the paper

Section II presents the novel automated response system to attacks developed in the RECLAMO project, where a special emphasis is placed upon deception responses according to the dynamic honeynets generated on virtualized platforms. We describe in Section III a framework for achieving an efficient DRAM, which is the main aim of an ongoing project called DHARMA. Finally, Section IV summarizes our contributions.

II. Virtual honeynets where diverting attacks

The main objective of RECLAMO [4] is the application of novel approaches for reacting to attacks, by means of defining, developing, and validating an intelligent AIRS able to conduct new and advanced reactions [6]. A special focus is taken on the so-called "deception-based" responses: diverting attacks to dynamically ad-hoc generated honeynets for being

adequately confined to mitigate them and learn from them. Thus, an intrusion is analyzed in real time by using a model of intrusions, responses, and security metrics that are formally defined with knowledge and behavior definition languages.

Fig. 1 illustrates the main functional blocks of RECLAMO, which are described next in the following subsections.

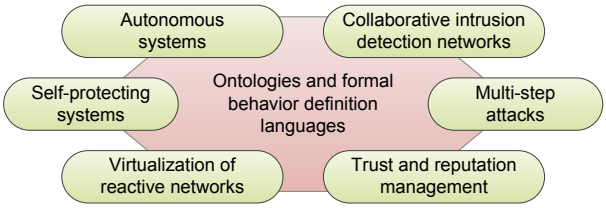

Fig. 1. Main functional blocks proposed in the RECLAMO project

The components belonging to each block defined in Fig. 1 are included within an envisaged architecture for RECLAMO, which are depicted in Fig. 2.

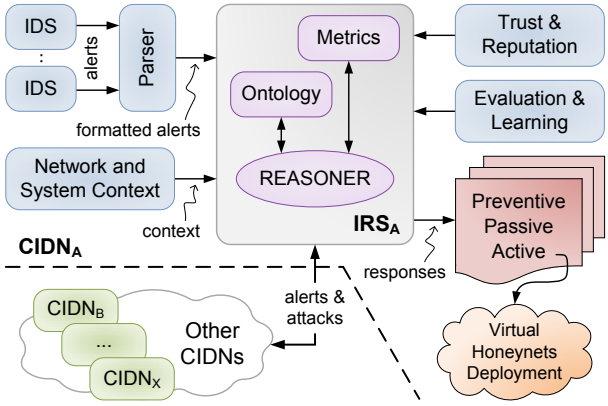

Fig. 2. System architecture of RECLAMO

As seen, concepts like autonomous system, ontologies, trust and reputation management, collaborative intrusion detection and prevention networks, self-protection as well as virtualized honeynets are clearly identified in Fig. 2. All of these concepts are considered as a key part of the novel automated response system to attacks proposed in RECLAMO. It is worth pointing out that all the related software of RECLAMO, regarding the components shown in Fig. 2, is publicly available at [4].

A. Autonomous systems applied to intrusion management

Autonomous computing systems are systems capable of managing themselves and dynamically adapting to changes, according to business rules and the objectives given by the system administrators. An autonomous computing system, or self-management system, has the following functions [7].

- *Self-configuration*. The system is able to immediately adapt to the deployment of new components or changes in the environment and configure itself automatically, without human intervention.
- *Self-healing*. The system is able to detect when, where, and why there are faults in the system, and carry out the appropriate fault-correction actions.
- *Self-protection*. The system detects hostile or intrusive behavior, such as unauthorized access attempts, virus, and denial of service attacks, and implements the appropriate actions to protect itself from them or cascading failures.

As shown in the previous definitions, the main feature of an autonomous system is the ability to specify their own behavior, in order to manage and protect itself and dynamically adapt to different conditions. The concept of self-protection, which the most important of any autonomous system, is the main component of the RECLAMO system, providing the ability to infer the most appropriate response for a given intrusion. This takes into account not just the intrusion, but also many other related parameters, such as the context or the trust and reputation of the network source. This autonomous system uses formally defined information models with ontologies for combining the intrusion information. Furthermore, it considers additional concepts such as self-evaluation learned parameters, trust and reputation of the different involved elements, and information coming from collaborative IDS/IPS systems in the same or different administrative domains.

There are several ontology languages, such as Ontolingua, KIF, OCML, OKBC, and F-Login, before the semantic web; and RDF, RDFS, DAML+OIL, OWL, and OWL2. The main ontology languages used in semantic web to formally describe information definitions are OWL and OWL2 [8]. Moreover, OWL is a knowledge definition language that structures the information into classes and properties, with hierarchies, and range and domain restrictions. However, the ability of OWL to define behavior into defined information is limited, so it is necessary to use additional rule languages like SWRL. This is the most widely used rule definition language, which extends the set of the OWL axioms by defining logical restrictions [9]. In RECLAMO, we use OWL and a set of SWRL rules.

Ontologies are the main semantic information model used within the scope of Semantic Web, Knowledge Management, and Artificial Intelligence. It formally represents a set of concepts, their meaning, and interrelation between them [10]. Initially, the propose of ontologies was to allow different and heterogeneous agents to share and reuse knowledge.

One of the main advantages when using ontologies is the formalization of the information semantics. This is important when dealing with heterogeneous information sources that can represent the same resource with different format and syntax. Another great advantage of using ontologies are the tools to define information and behavior, improving the usage of ontologies and rule languages in several environments.

As a first task, it is required to design a formal definition about vulnerabilities, attacks, and response actions based on different taxonomies. These ontologies define a subset of vulnerabilities, attacks, and responses and allow specifying the behavior of the autonomous system when an intrusion is detected. So that, the system can reason and infer new knowledge from different inputs.

The AIRS is able to understand heterogeneous alerts and to know whether they are referring to the same intrusion or not. Nowadays, there are several data format standards for alerts representation, such as IDMEF (Intrusion Detection Message Exchange Format) [11]. This is defined in XML to represent, exchange, and share the information about intrusion detection, but with no additional knowledge representation. IDMEF can be useful for the AIRS in order to correlate alert information with other additional data, such as network context and rules. RECLAMO uses ontology mapping technologies, such as D2RQ [12], which allows mapping between relational databases and OWL/RDFS ontologies.

B. Autonomous intrusion response systems: Self-protecting

AIRSs are security technologies to trigger dynamic reaction against detected intrusions. The system infers the mot suitable response and triggers it automatically without participating an administrator. The state of the art in AIRSs is not as mature as with IDSs, although several systems have been proposed in recent years: AAIRS, ADEPTS, EMERALD, CSM, FAIR, and IDAM&IRS. For example, Stakhanova presented in [13] a taxonomy of autonomous intrusion response systems, together with a review of current trends in intrusion response research. According to this paper, AIRSs can be classified in different ways according to various features:

- *By ability to adjust*: static and adaptive. The response selection mechanism remains the same during the life of the AIRS in static AIRSs. Adaptability is a powerful feature that can automatically modify the chosen response according to other external factors, like the previous response effectiveness or changes in the environment.
- *By response selection mechanism*: static, dynamic, and cost-sensitive mapping. There is an increasing interest in recent years in developing cost-sensitive models for response selection. The primary objective of these models is to ensure an adequate response without sacrificing the normal system functionality. That is to say, the system takes into account the complexity and cost of the reaction, besides the impact of the intrusion.
- *By time of response*: proactive and delayed. Proactivity is the ability of the AIRS to react against an intrusion or attack before it takes place. A reactive AIRS infers and activates the reaction when the intrusion is detected.
- *By response cost model*: static, static evaluated (S), and dynamic evaluated cost (D). This refers to the evaluation mechanism that the AIRS uses to get the response cost.

To achieve an optimal response in the shortest time, it is required that the AIRS is adaptive, cost-sensitive mapping, and proactive. But there is another feature, the *semantic coherence*, that is not present in this taxonomy and it is especially crucial in a heterogeneous intrusion detection environment.

In TABLE I, we show the features of some related AIRSs, where only ADEPTS and the Stakhanova's IRS offer adaptive, proactive, and cost-sensitive functionalities. However, none of them provides mechanisms to archive semantic coherence. Due to this, the AIRS proposed by RECLAMO supports semantic coherence by using ontologies, formal behavior specification languages, and reasoning mechanisms, as well as fulfilling the rest of requirements. Moreover, our system includes a formal mechanism to evaluate the cost of responses, giving feedback to the system for improving responses in future attacks.

	AAIRS	ADEPTS	CSM	EMERALD	Stakhanova's	FAIR	IDAM&IRS	Network IRS
Adaptive	√	√			√			
Proactive		√	√		√			
Cost-sensitive		√			√	√	√	√
Evaluated cost	S				S	S	S	D
Semantic coherence								

TABLE I
FUNCTIONALITIES OF EXISTING AIRS

The use of ontologies and formal languages, so as to define the behavior of the autonomous system, is essential to provide AIRSs with the self-protection capability, and so fixing the problem of semantic coherence. Due to its expressiveness and flexibility, they also enable AIRSs to meet other requirements: adaptability, proactivity, and response cost-sensitive.

Another key feature to take into account is the *cooperation* capability: autonomous and cooperative. Network-based IRSs are often built in such a cooperative way, because they provide more effective responses than single and autonomous systems. An example of it is EMERALD. Due to this, the RECLAMO project follows the cooperative way.

Finally, the autonomous response system is combined with other technologies, e.g, the correlation of the alerts, as well as further information such as the trust and reputation of the IDSs together with their network context [14]. Therefore, the response system of RECLAMO is capable of detecting attacks in a given organization, and reacting against them quickly in an autonomous and optimal way, with no intervention from network administrators. The reaction includes the inference of the optimal response in a diagnosis phase, and the deployment of that response in a reaction phase.

C. Collaborative intrusion detection systems

Collaborative intrusion detection systems (CIDS) emerged in recent years to deal with detecting distributed attacks, where pieces of evidence are gathered at different network locations to be subsequently correlated. This distributed nature of attack execution is due to the way in which attackers perform their malicious practices, evolving toward a new mode of operation more global and distributed. In case a collaborative strategy is not used, alerts generated by the IDSs are viewed as isolated incidents, with no relevance when analyzed separately [2]. Due to this, alerts should be treated as a whole from a more global viewpoint for knowing the actual state of the network, by detecting distributed attacks after deploying multiple IDS instances among security domains. A partnership of all IDSs

will form an overlay layer for sharing security data among peers: alerts and incidents detected by each IDS individually. This cooperative system is a *Collaborative Intrusion Detection Network* (CIDN) that allows building a collective knowledge base of isolated alerts within a given security domain [15], whereas the union of several CIDNs shapes a *Collaborative Alert System* (CAS) to detect attacks more distributed among several security and/or administrative domains.

Where to place the pool of IDSs is a key point for sharing alerts properly among them. A survey on how the IDSs can be distributed is given in [2], which is focused on centralized, decentralized, and distributed architectures. Instead, we think that a partially-decentralized approach is the best placement model to tackle the drawbacks implicit in the other. Partially-decentralized schemes address the problems of having a single point of failure and the lack of scalability, from centralized approaches; and the overhead and management difficulty, from decentralized and distributed schemes. A schematic example of the system architecture of RECLAMO is shown in Fig. 3, which is based on a partially-decentralized scheme.

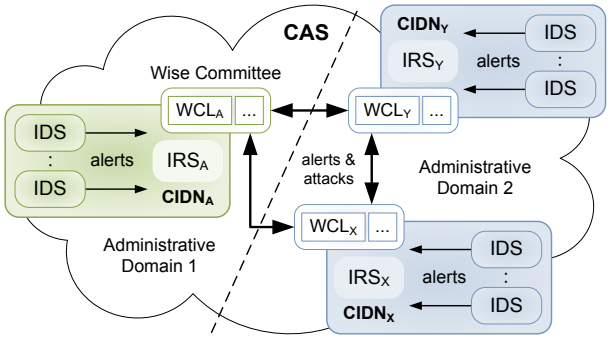

Fig. 3. Partially-decentralized scheme of a CAS

This partially-decentralized scheme is built with the help of a pool of *supernodes* (or *superpeers*) that act as the heads of their security domain, thereby shaping a *Wise Committee* (WC), one per CIDN. They are in charge of assessing the alerts generated by the IDSs before being finally shared with the rest of the CIDN's IDSs (intra-domain knowledge base) or other administrative domains at the CAS level (inter-domain knowledge base) [16]. Sharing the internal knowledge base of alerts of a CIDN with other CIDNs is carried out by the most trustworthy IDS of the CIDN, called *WC Leader* (WCL).

D. Trust and reputation management

CIDSs assume that the IDSs cooperate each other honestly to correlate security incidents: alerting when a threat occurs or not alerting when the system is safe. However, honesty may produce a misleading perception on the security state in case the IDSs exhibit a malicious behavior, reporting bogus alerts to provoke errors to other IDSs. Identifying malicious attitudes can be achieved by using trust and reputation mechanisms, with which to model the behavior of the IDSs [17].

Let us suppose that a given IDS j wants to share a new alert with the rest of the CIDN members for correlation purposes. This IDS sends the alert to the WC for being assessed before its publication to the rest of IDSs. The WC members compute the j's reputation, $Rep(j)$, to decide if the alert is a true or false positive depending on that reputation score. To this end, the WCL only queries recommendations to those IDSs with similar detection skills than j; otherwise, the IDSs could not know the satisfaction on the alerts detected by j. These recommendations will be finally aggregated and shared by the WCL with the rest of the WC members, where each WC_i will assess its trust on j, $T_{WC_i}(j) \in [0,1]$, as given in (1).

$$T_{WC_i}(j) = \left(\bigoplus_{\substack{k=1,\\ k \neq i}}^{|IDS|} \omega_{i,k} \cdot Rep_i(k) \cdot Rec_k(j)^{\varphi} \right) \quad (1)$$

where $|IDS|$ is the total number of IDSs in the CIDN; $\bigoplus$ defines an aggregation operation chosen by the administrator; $Rep_i(k) \in [0,1]$ is the reputation of k from the perspective of WC_i; $Rec_k(j) \in [0,1]$ defines the recommendation value gathered from k on j; $\varphi \geq 1$ represents the pace at which WC_i "forgets" recommendations; and $\omega_{i,k} \in [0,1]$ is the weight that WC_i can deposit on the type of IDS that k represents. For example, $\omega_{i,k}$ may be divided into separate three weights according to the k's group: α_i for NIDSs, β_i for HIDSs, and φ_i for the WC's NIDSs, with $\alpha_i + \beta_i + \gamma_i = 1$.

The recommendation element is a key factor to assess alert satisfaction. The recommendation of IDS k on IDS j at a given time t is computed by (2), taking into consideration the previous recommendation values already computed by k.

$$Rec_k^{(t)}(j) = \upsilon_k \cdot Rec_k^{(t-1)}(j) + (1 - \upsilon_k) \cdot Sat_k^{(t)}(j) \quad (2)$$

where $\upsilon_k \in [0,1]$ is the weight to previous recommendation values and $Sat_k(j) \in [0,1]$ represents the satisfaction of IDS k on the alert published by j, according to the configurations declared by IDSs k and j in their bootstrapping phase. $Sat_k(j)$ may vary by several factors, depending on whether k has direct or indirect evidences about the alerts published by j.

The configuration of both IDSs comes into play to determine possible evidences in assessing alert satisfaction. First, if both IDSs are deployed in the same network, the alert shared by j would also have been produced by k, provided that the they are implementing similar detection skills. In case k has produced an alert to warn the same incident, the satisfaction of k on j would be the highest according to the j's reputation (from the perspective of k) and the alert severity.

Secondly, when the two IDSs are deployed in the same network, but with different configurations in detection, k has to infer the detection skills that j used to produce the alert when the former did not generate it. To this end, IDS k can base its decision on the detection skills of other IDSs in its same network and their attitude with respect to the alert produced by j (indirect experience-based approach). Specifically, k can follow a majority-based voting scheme, taking into account

both the configurations implemented by the other IDSs that detected or not the same incident than j, in comparison with the configuration of k, and the reputation score of each.

Finally, when both IDSs are deployed in separate networks, there is no actual obligation on k to alert about the incident detected by j. Yet, it is possible that k had to detect it in case a number of its neighbors in this same network did detect it. In this case, IDS k should only compute the new recommendation value of IDS j as defined in (2).

As final step to compute $Rep(j)$, the reputation on IDS j is computed by aggregating all partial trust values given by (1), calculated by every WC_i. This aggregation process is carried out by the WCL through computing (3).

$$Rep(j) = \bigoplus_{i=1}^{|WC|} T_{WC_{i,j}} \quad (3)$$

Depending on the j's reputation score, denoted as $Rep(j)$ in (3), the alerts shared by j are finally published by the WC to the rest of IDSs of the j's CIDN. Furthermore, the WCL will send these alerts to other WCLs of the CAS if the alerts' severity denotes a potential evidence of a distributed attack, determined by the severity of such alerts. Information sharing in a distributed environment requires another trust and reputation mechanism aimed to work at an inter-domain level. In this context, the WC that receives an alert from another domain has to assess the reputation of the latter in order to decide whether to accept or not the alert and share it with its members. For this purpose, we adapt the application of a well-known trust model such as PeerTrust [18].

PeerTrust is a reputation-based trust supporting framework that includes a coherent adaptive trust model for quantifying and comparing the trustworthiness of the entities according to a transaction-based feedback system. Thus, it fits well for an inter-domain reputation mechanism. PeerTrust introduces three basic trust parameters and two adaptive factors in computing trustworthiness of entities, as given by:

$$T(\Omega) = \alpha \cdot \sum_{i=1}^{I(\Omega)} S(\Omega, i) \cdot Cr(p(\Omega, i)) \cdot TF(\Omega, i) + \beta \cdot CF(\Omega)$$

where α and β denote the normalized weight factors for both the collective evaluation and community context factor, respectively; $I(\Omega)$ represents the total number of transactions performed by Ω with other domains; $S(\Omega, i)$ is the normalized amount of satisfaction that Ω receives from $p(\Omega, i)$ in its i-th transaction; $Cr(p(\Omega, i))$ denotes the credibility of the feedback submitted by other domains; $TF(\Omega, i)$ represents the adaptive transaction context factor for the i-th transaction of Ω; and $CF(\Omega)$ is the adaptive community context factor for Ω.

E. Definition of security metrics and behavior of the system

Existing AIRSs make use of several fixed response metrics to choose the action that the system must execute, such as the ones used by AAIRS, ADEPTS, CSM, EMERALD, the Stakhanova's IRS, FAIR, and IDAM&IRS. All the response metrics allow the AIRS to choose the reaction that may trigger, but they are fixed and cannot be dynamically chosen (i.e., the AIRS always uses the same metric, regardless of the intrusion context or the state of the system).

We defined in RECLAMO a pool of response metrics for modeling and specifying the behavior of the AIRS [19]. The knowledge included in the ontology allows inferring the most appropriate response set for different events. Such metrics are defined by using a formal language of behavior specification, so the complete AIRS behavior is defined formally. Their specification in a flexible and dynamic way requires the use of a specific language able to express these metrics. We use SWRL as a formal language to express them.

The following parameters have been identified as the most relevant ones for inferring the optimum response: the intrusion impact, the IDSs' confidence, the importance of the affected resources, the severity and cost of the response, and its success. Regarding the relevance that the compromised resource has for the organization, the AIRS assigns more or less weight to each of these parameters. For example, consider that the affected resource is a user workstation. The response cost may take priority over its success rate. However, if the attacked resource is a database server, the AIRS may give more importance to the response severity and the response effectiveness than the high cost of executing the response.

In order to choose the responses, three response metrics have been taken into account in RECLAMO, which are next presented. Each metric assigns a weight to the parameters in a different way. Depending on the level of importance of the resource, the system applies one metric or another.

1) Damage reduction metric: The purpose of this metric is to strike a balance between the cost of the damage caused by an "unattended" attack (the intrusion impact) and the cost of deploying the response (the response impact). This metric aims to avoid that the response has a greater negative effect than can cause the attack on the resources of an organization (e.g., loss of availability of several resources). The AIRS uses this metric regardless of the component importance. Note that this metric is equivalent to the one defined by Stakhanova.

The application of this metric infers the responses, whose impact is lower than or equal to the product of the intrusion impact and the IDS confidence, as defined in (4).

$$Impact_{intrusion} \cdot Confidence_{IDS} > Impact_{response} \quad (4)$$

Then, the AIRS discards the responses whose impact is greater than the intrusion impact. Note that his metric depends on three parameters: the intrusion impact, the IDS confidence, and the response impact. The AIRS does not compute these parameters at inference time. They correspond to the properties of the defined ontology, and their values are input to the response system which must have previously defined.

2) Minimum cost metric: The AIRS applies the minimum cost metric when the affected component is not very relevant for the organization. The purpose with this metric is thus to minimize the response total cost, as given by (5), so that the AIRS will trigger the execution of the lower cost response.

$$Cost_T\ response = Impact_{response} + Cost_d\ response \quad (5)$$

The response total cost includes the response impact and the response deployment cost. The former, $Impact_{response}$, represents the cost that executing the response involves to the organization, in terms of the damage that the response action causes to the resources of the organization. The latter, $Cost_d\ response$, represents the cost that the deployment of the response involves to the organization, in terms of the required resources (e.g., number of the needed routers or the number of backups). The lower cost responses are usually the lower complexity responses. Thus, if several responses have the same cost, the AIRS will select the lower complexity response.

3) Highest severity and highest efficiency metric: When the compromised resource is critical, the response system uses the highest severity and efficiency metric, whose purpose is to maximize the response severity and the success. This metric depends on the results of the previous executions of the specific response against a given similar intrusion, the severity associated with the intrusion, and the severity of the response itself. Thus, the purpose is to satisfy (6) and maximize (7).

$$RE \cdot |Severity_{response}| > Severity_{intrusion} \quad (6)$$

$$\max\left(RE \cdot |Severity_{response}|\right) \quad (7)$$

As shown in (6) and (7), the metric depends on:

- *Intrusion severity*, $Severity_{intrusion}$. The AIRS gets the intrusion severity according to some of the predefined equivalences between intrusion type and severity.
- *Response absolute severity*, $|Severity_{response}|$, whose value is previously set by the system administrator.
- *Response efficiency*, RE. It measures the success of the previous response against an intrusion. Partial efficiency of the response is calculated after each execution of it, which is based on machine learning methods [20]. These methods analyze all the data captured from the context information (network and system context).

F. Virtualized honeynets as a response system

The reaction phase comes after the diagnosis is completed. Classical reactions typically consist on deploying new firewall rules, whereas most recent AIRSs also offer new possibilities to react against attacks; for example, by creating honeynets. These honeynets can be specifically adapted to the detected attack. In this context, the RECLAMO project proposes a reaction based on the configuration and automatic deployment of honeynets, which are optimized and adapted to each attack according to the specifications provided by the AIRS.

A honeynet is defined as a set of honeypot systems (servers, routers, switches) prepared to be attacked, while monitoring the attack simultaneously. A honeypot can have low- or high-interaction. As opposite to the low-interaction honeypots that just emulate operating systems and services, high-interaction honeypots provide real systems, application, and services to lure attackers. A honeynet can consist of high-interaction or low-interaction honeypots, or both of them. A honeynet creates a highly controlled network, with which the administrators can control and monitor all the activity that occurs inside it.

Given the complexity involved in launching a honeynet, the virtualization techniques are an essential tool that greatly facilitates its deployment and management. These tools can create multiple logical systems with a single physical machine, thereby drastically reducing the number of physical systems required so as to create a honeynet. These honeynets are built ad-hoc and optimized in order to get as much information as possible from each attack. This dynamic honeynet generation uses advanced virtualization techniques capable of generating large scale heterogeneous honeynets.

Virtualization tools facilitate the definition of the honeynets, including their topology, addresses, system type, deployment, and monitoring. So, they hide the complexity of the underlying virtualization platforms to the final system. Among these tools available in the market, we can find VNX, VNUML, Netkit, MLN, and vBET. In RECLAMO, we chose Honeyd and VNX as the frameworks able to deploy low-interaction honeypots and high-interaction honeypots [21]. VNX includes the ability of deploying a virtual network scenario over cluster of servers, improving the scalability of the solution and also allowing the creation of very complex honeynets, even over distributed cluster infrastructures [22].

There are several projects and initiatives that have made use of virtualization as a basic tool with which to dynamically create honeynets. One of the most advanced is Collapsar [23], which combines a powerful distributed traffic capture system with a server farm, where the interesting traffic is redirected to dynamically create honeynets that process it.

The dynamic generation of the honeynets require a previous characterization and parametrization of different honeynets. The objective is to generate a large and flexible *catalog of honeynets* for being used by the AIRS. Each of these honeynets can be tuned at deployment time for its customization with the aim of facing specific attacks. This is a key feature since it provides flexibility to the system, which allows executing the traditional scenarios with static honeynets as well as advanced scenarios built for the autonomous system dynamically.

III. Toward the dynamicity for risk assessment and management

The RECLAMO project presented in Section II is capable of reacting to a given input (an intrusion), and getting additional context and collaborative information with the aim of inferring the most appropriate response. But this approach is static and reactive: reacting when there is an intrusion and processing the context information acquired in that given moment. Thus, the architecture proposed in RECLAMO (Fig. 2) can be enhanced with a dynamic and proactive approach, this being the main objective of the DHARMA (*Dynamic Heterogeneous threAts Risk Management and Assessment*) project [5].

DHARMA becomes possible to separate the concepts of dynamic risk assessment and the consequences after assessing that risk. This latter can be an automated response triggering, like the one proposed in RECLAMO, or any other output, such as dynamic risk visualization in a control panel, updating of risk assessment methodologies or proactive actions.

The major difference between these projects is that the main input in RECLAMO is the intrusion, being the context processed just to enrich the inference process for it, whereas the intrusion in DHARMA is just another input, and context changes play a principal role for a dynamic risk assessment. So, there will be a constant evaluation of the heterogeneous context parameters in order to assess any change in the risk level, as stated by the organization. The main components of DHARMA are detailed in the next sections, which are defined within the envisaged architecture shown in Fig. 4.

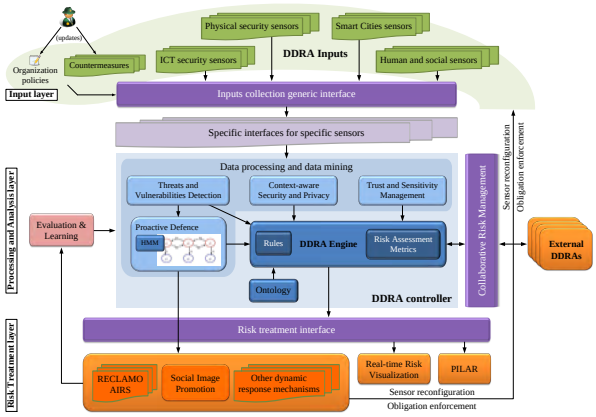

Fig. 4. Technical architecture of DHARMA proposal

A. Heterogeneous information monitoring mechanisms

The inclusion of new information from different sources is a key feature for the risk analysis process, which must take into account the parameters involved in this process as well as those already used in traditional risk analysis such as assets, threats, vulnerabilities, and countermeasures. These new sources are grouped into four main set of sensors: physical security, ICT security, Smart Cities, and human and social sensors. All these sensors are represented in the top layer of Fig. 4.

The management of the heterogeneous information allows correlating and processing information from multiple and distributed sources, through data mining techniques, in order to detect any alteration in the monitored parameters as well as find out malicious attempts on assets. As a consequence, the dynamic risk assessment controller will capable of taking the required actions as a response from a proactive defense model, by using algorithms of machine learning such as neuronal networks and Hidden Markov Models.

Sharing information between sensors and the (distributed) dynamic risk assessment controller also requires the definition of trust and sensitivity management models, similarly to the one presented in Section II-D. They have to quantify the actual threat level on the assets, according to the trustworthiness of the subjects accessing to the assets (threat grows up as the subject's trust score decreases) and the sensitivity of the resources, e.g., personal data, contained in the asset (threat grows up as the asset's sensitivity level increases).

B. Distributed and dynamic risk assessment and management

Risk assessment and management (RA/RM) is a common and extended practice used in the most of large corporations to identify, analyze, and either accept or apply countermeasures to mitigate the risk –expected likelihood of unfortunate events and the impact of these events on the assets of the organization. A lot of mature methodologies, techniques, standards, and have been specified, developed, and implemented till the date, such as MAGERIT, ISO 27005:2011, OCTAVE, CRAMM, EBIOS, NIST SP800-30, COBRA, and RA2, among others. They have been widely evaluated and tested using well defined metrics as well as benchmarking schemes.

But now, with the current dynamicity and heterogeneity of the security threats area, there is a large need for dynamic risk assessment and management processes that allow systems to update the level of risk at real time, as well as dealing with the dynamicity and ever changing nature of the technology and threats. The traditional approaches of risk management and assessment do not cover this need because it is not a continuous process, but the assessment process is repeated regularly over discrete and large time intervals. Although these intervals were smaller, they leave a window of opportunity where assets could be affected.

A new approach, known as Dynamic Risk Assessment and Management (DRA/DRM), seems to be the solution to the new need and its aim is to continuously update the risk of any changes happening in the organization: changes in threats, new vulnerabilities, new countermeasures or modification of assets. In the scope of dynamic frameworks for risk assessment and management, some efforts have been done, but none of the proposed work is sufficiently mature, and the usage of new sensors of different nature (such as environmental sensors) as inputs to the risk analysis could improve the accuracy and efficiency of the management process.

The DHARMA architecture, depicted in Fig. 4, relies on the design and integration on different heterogeneous sensors that continuously monitor different sources that might trigger a potential threat to the organization. All these sensors are not limited to the traditional ICT security incidents, but also to many other sources: environmental sensor (threats coming from changes in temperature, humidity, etc.), physical sensor (threats coming from physical presence and/or recognition), vulnerabilities sensor (threats coming from new vulnerabilities that might affect the organization assets), and also sensors trying to evaluate potential threats to the organization raised from the social networks activity (social networks sensor) or even from the own organization employees (human resources sensor that tries to evaluate the level of labor disputes and conflicts in the organizations). The continuous monitoring of these highly dynamic context parameters is the key for an accurate dynamic risk assessment methodology.

C. Deployment and enforcement of dynamic countermeasures

The results of the risk analysis process can be treated from different approaches: passive, preventive or proactive, reactive, and collaborative. The deployment and enforcement

of response actions, when needed, requires the definition of mechanisms with which to treat the computed impact and risk. They can incorporate the definition of interfaces to the existing AIRSs, such as the one presented in Section II; the deployment of social image promotion actions to revert possible harms in the market on the organization's corporate image after being victim of a threat; and certain mechanisms enabling the reconfiguration of sensors as needed to maximize the information gain [24], depending on the feedback provided by the distributed dynamic risk assessment controller. In the former case, the deployment of dynamic countermeasures as response actions in protecting assets allows inferring the optimum reaction as a defense strategy, diverting the attack to a honeynet where to mitigate it and learn from it.

The deployment and enforcement of countermeasures, with the aim of protecting the assets according to the current risk level, is shown in the lower layer of Fig. 4.

IV. Conclusion

We have reviewed throughout this paper the main objectives and relevant topics of two R&D projects. First, the RECLAMO project deals with important key technologies like autonomous systems and trust and reputation management, and combine them in a single solution to provide an automated response system to attacks. As a promising response for RECLAMO, we developed the dynamic generation and deployment of honeynets where the attacks are diverted for isolation.

The proposal of RECLAMO has been extended to manage the current dynamicity and heterogeneity present in current systems, due to the large number of heterogeneous sensors, reporting threats who exploit vulnerabilities on the assets, and contextual changes in the organization. This new challenge is being addressed in an ongoing R&D project called DHARMA. Its main goal is to provide assistance for dynamic assessment and management of the risk, and dynamically reassessed it in real time in order to prevent, react, and mitigate potential threats on sensitive assets of an organization.

Acknowledgment

This work has been partially funded with the support from the Spanish MICINN (project RECLAMO, *Virtual and Collaborative Honeynets based on Trust Management and Autonomous Systems applied to Intrusion Management*, with codes TIN2011-28287-C02-01 and TIN2011-28287-C02-02, and project DHARMA, *Dynamic Heterogeneous Threats Risk Management and Assessment*, with codes TIN2014-59023-C2-1-R and TIN2014-59023-C2-2-R) and the European Commission (FEDER/ERDF). Thanks also to the Funding Program for Research Groups of Excellence granted by the Séneca Foundation with code 04552/GERM/06.

References

[1] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and big heterogeneous data: A survey," *Journal of Big Data*, vol. 2, no. 1, pp. 1–41, Dec. 2015.

[2] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and survey of collaborative intrusion detection," *ACM Computing Surveys*, vol. 47, no. 4, pp. 55:1–55:33, Jun. 2015.

[3] M. Gil Pérez, V. Mateos Lanchas, D. Fernández Cambronero, G. Martínez Pérez, and V. A. Villagrá, "RECLAMO: Virtual and collaborative honeynets based on trust management and autonomous systems applied to intrusion management," in *Proceedings of the 7th International Conference on Complex, Intelligent, and Software Intensive Systems*, Jul. 2013, pp. 219–227.

[4] Universidad Politécnica de Madrid and Universidad de Murcia, "Web site of the RECLAMO project," [Online] http://reclamo.inf.um.es.

[5] Universidad Politécnica de Madrid and Universidad de Murcia, "Web site of the DHARMA project," [Online] http://dharma.inf.um.es.

[6] V. A. Villagrá González and G. Martínez Pérez, "RECLAMO: Red de sistemas de engaño virtuales y colaborativos basados en sistemas autónomos de respuesta a intrusiones y modelos de confianza," *Revista SiC: Ciberseguridad, Seguridad de la Información y Privacidad*, no. 111, pp. 64–65, Sep. 2014.

[7] F. T. M. Bazier, J. O. Kephart, H. Van Dyke Parunak, and M. N. Huhns, "Agents and service-oriented computing for autonomic computing," *IEEE Internet Computing*, vol. 13, no. 3, pp. 82–87, Jun. 2009.

[8] W3C Recommendation, "OWL 2 Web ontology language document overview (2nd edition)," Dec. 2012.

[9] W3C Member Submission, "SWRL: A semantic web rule language combining OWL and RuleML," May 2004.

[10] S. Staab and R. Studer, *Handbook on ontologies*. Springer Science & Business Media, 2013.

[11] H. Debar, D. A. Curry, and B. S. Feinstein, "The intrusion detection message exchange format (IDMEF)," IETF RFC 4765, Mar. 2007.

[12] R. Cyganiak, C. Bizer, J. Garbers, O. Maresch, and C. Becker, "The D2RQ mapping language," [Online] http://d2rq.org/d2rq-language.

[13] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," *International Journal of Information and Computer Security*, vol. 1, no. 1/2, pp. 169–184, Feb. 2007.

[14] J. J. Martínez Molina, M. A. Hernández Ruíz, M. Gil Peréz, G. Martínez Pérez, and A. F. Gómez Skarmeta, "Event-driven architecture based on patterns for detecting complex attacks," *International Journal of Critical Computer-Based Systems*, vol. 1, no. 4, pp. 283–309, Nov. 2010.

[15] C. Fung, "Design and management of collaborative intrusion detection networks," Ph.D. dissertation, University of Waterloo, Canada, 2013.

[16] M. Gil Pérez, F. Gómez Mármol, G. Martínez Pérez, and A. F. Skarmeta Gómez, "RepCIDN: A reputation-based collaborative intrusion detection network to lessen the impact of malicious alarms," *Journal of Network and Systems Management*, vol. 21, no. 1, pp. 128–167, Mar. 2013.

[17] M. Gil Pérez, F. Gómez Mármol, G. Martínez Pérez, and A. F. Skarmeta Gómez, "Building a reputation-based bootstrapping mechanism for newcomers in collaborative alert systems," *Journal of Computer and System Sciences*, vol. 80, no. 3, pp. 571–590, May 2014.

[18] L. Xiong and L. Liu, "PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 843–857, Jul. 2004.

[19] V. Mateos Lanchas, V. A. Villagrá, F. Romero, and J. Berrocal, "Definition of response metrics for an ontology-based automated intrusion response systems," *Computers & Electrical Engineering*, vol. 38, no. 5, pp. 1102–1114, Sep. 2012.

[20] P. Holgado, V. A. Villagrá, and V. Mateos Lanchas, "Redes neuronales aplicadas al proceso de aprendizaje de un sistema de respuestas a intrusiones automático," in *XI Jornadas de Ingeniería Telemática*, Oct. 2013, pp. 419–426.

[21] F. Wenjun, D. Fernández, and V. Villagrá, "Tecnology independent honeynet description language," in *Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development*, Feb. 2015, pp. 303–311.

[22] F. Galán, D. Fernández, J. E. López de Vergara, and R. Casellas, "Using a model-driven architecture for technology-independent scenario configuration in networking testbeds," *IEEE Communications Magazine*, vol. 48, no. 12, pp. 132–141, Dec. 2010.

[23] X. Jiang and D. Xu, "Collapsar: A VM-based architecture for network attack detention center," in *Proceedings of the 13th USENIX Security Symposium*, Aug. 2004, pp. 15–28.

[24] M. Gil Pérez, J. E. Tapiador, J. A. Clark, G. Martínez Pérez, and A. F. Skarmeta Gómez, "Trustworthy placements: Improving quality and resilience in collaborative attack detection," *Computer Networks*, vol. 58, pp. 70–86, Jan. 2014.