universidad de león

RIaSC

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

Neural Networks applied to the learning process of Automated Intrusion Response systems

Pilar Holgado, Víctor A. Villagra
Departamento de Ingeniería y Sistemas Telemáticos, Universidad Politécnica de Madrid Avenida Complutense, 30, 28040, Madrid
pilarholgado@dit.upm.es, villagra@dit.upm.es

***Abstract*- The architecture for an Automated Intrusion Response System (AIRS) has been proposed in the RECLAMO project. This system infers the most appropriate response for a given attack, taking into account the attack type, context information, and the trust of reports from IDSs. Also, it is necessary to evaluate the result of previous responses, in order to get feedback for following inferences. This paper defines an algorithm to determine the level of success of the inferred response. The objective is the design of a system with adaptive and self-learning capabilities. Neural Networks are able to provide machine learning in order to get responses classification.**

I. INTRODUCTION

Security is an important issue for any corporation. They have to keep their systems safe from external attacks to maintain the service levels. Also, they must provide to costumers inside information and correct operation of applications.

As the number of security incidents increases, becoming more sophisticated and widespread [1], Intrusion Detection Systems (IDSs) [2] have evolved rapidly and there are now very mature tools based on different paradigms (statistical anomaly-based [3], signature-based and hybrids [4]) with a high level of reliability. IPSs (Intrusion Prevention Systems) have also been developed by combining IDS with a basic reactive response, such as resetting a connection. IRSs (Intrusion Response Systems) leverage the concept of IPSs and provide the means to achieve specific responses according to some predefined rules.

Nowadays, IRSs are playing an important role in the security architecture. These systems mitigate the impact of attacks in order to keep integrity, confidentiality and availability of the resources. Automated Intrusion Response Systems (AIRS) provide the best possible defense, as well as shortening or eliminating the delay before administrators come into play.

AIRSs are security technologies with the goal of choosing and triggering automated responses against intrusions detected by IDSs, in order to mitigate them or reduce their impact [5].

Metrics are defined to measure different parameters necessary for response selection, such as the IDS confidence, the network activity level, the reliability of intrusion reports, and the importance of network components. Also, it is very relevant taking into account the complexity, severity, cost and efficiency of responses.

Current AIRSs have a fixed approach to response metrics, so that the metric cannot be dynamically chosen. We propose a security architecture to be able to dynamically select the most appropriate response. It takes into account factors such as the systems context, cost of responses, importance of resources, and efficiency of responses.

In this scope, the RECLAMO project (Virtual and Collaborative Honeynets based on Trust Management and Autonomous Systems applied to Intrusion Management), an R&D project funded by the Spanish Ministry of Science and Innovation, defines an AIRS able to dynamically interpret metrics. In particular, it is achieved with an adaptive autonomous system based on assessment of the harm caused for the intrusion and the cost of the responses.

The autonomous system is developed based on information formal models defined by ontologies [6]. It is used to represent intrusion information, parameters of self-evaluation, confidence and reputation of IDSs, and others. The security metrics use this information to infer the most appropriate response. These metrics are represented in the formal language SWRL (Semantic Web Rule Language) [7].

IDMEF-based Ontologies (Intrusion Detection Message Exchange Format) [8] are used to homogenize information. IDMEF format provides a common language to generate alerts about suspicious events and are stored in Ontology classes [9]. The Ontology have been defined using OWL (Web Ontology Language) [10]. It takes advantages of Semantic Web, such as information inference.

Our goal is to analyze the response efficiency triggered after the arrival of an intrusion. Moreover, we want the AIRS to automatically learn from previous responses. Specifically, we propose the use of Neural Networks for this task.

Artificial Neural Networks are a branch of Artificial Intelligence called Machine Learning. In this group, there are a lot of learning techniques [11]. The purpose of our algorithm is to classify the success of the triggered response. Thus, the AIRS is able to accomplish a self-learning process through a response rate for future incidences of same kind.

The remainder of this paper is organized as follows. Section II outlines the current state of the art in evaluation response and Neural Network. The architecture of the AIRS is presented in section III. Section IV gives an overview of the inference process of the response system. The Neural Network proposed is shown in section V along with its topology and parameters.

The mathematical algorithm of artificial neural network is detailed in section VI. Section VII explains how to calculate the response efficiency and finally, conclusion and future works are included in section VIII.

II. RELATED WORKS

Nowadays, Automatic Learning is a key concept in the prevention, detection and response systems against intrusions. In this section we show some current research in response effectiveness.

MAIM [12] is an adaptive intrusion response system based in artificial immune. This approach implements a policy according to global risk, rather than focusing on individual attacks. Like us, they use a bio-inspired algorithm. Specifically they are based on the immune system. They learn about the dangerous states of the network to detect false positives. Then a more or less strict response to an attack based on predefined policies is triggered. In contrast, we use bio-inspired learning for assessment of triggered response to a given attack. In addition, we perform context analysis for false positives detection and we launched a more or less strict response based on the attack and response costs in terms of the significance of the assets.

In [13], multi-step attacks are mitigated through several executions of responses sets. At the end of each response set, a mechanism that measures the effectiveness of such responses is launched. In particular, the online risk assessment measures the risk index of an applied round of responses instead of one applied response. This affects the order in which the responses are triggered within the set in a determinate level. In our case, the measurement of response effectiveness is entirely individual based on system and network contexts.

COSIRS [14] is based in three factors for response assessment: cost of intrusion damage, cost of automatic response and operational cost. In contrast with the system proposed in this paper, they have not taken into account the effectiveness against the attack in progress. That is, whether the selected response has been successful against intrusion at this time. Moreover, we have also taken into account in our metrics the intrusion cost, response cost and deployment cost [18].

Also, neural networks have been used in network security, specifically, there are approaches using supervised and unsupervised algorithms in intrusion detection systems. For example, [15] and [16] use Backpropagation algorithm for IDSs.

III. ARCHITECTURE

Fig. 1 represents the modules of AIRS proposed in [17]. The goal of this architecture is to choose the optimal response of a set of available responses.

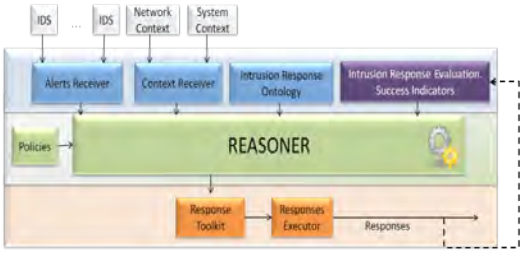

Fig. 1. Architecture of AIRS based Ontology

The AIRS receives a set of inputs including the intrusion reports, context information, policies of security metrics and intrusion response ontology. The policies specify different metrics that will be chosen depending on the context and type of intrusion.

The *Reasoner* runs inference process to choose the best response based in other modules (Policies, Alerts Receiver, Context Receiver and Intrusion Response Ontology). OWL is used to define all the information of the response process.

The *Intrusion Response Ontology* defines the concepts and relationships needed in the autonomous system. The ontology is based on the IDMEF structure including classes and properties. There are two main classes related with the evaluation system: *Response* and *Result*. Each of these classes have properties related with this objective. The *Response* class has two properties associated, *executionTimes* and *sucessFactor*. *responseEfficiency* is the property included in the *Result* class. They are defined in section VII.

The *Intrusion Response Evaluation* module evaluates the responses triggered by the AIRS and is the scope of the main contribution of this paper. We propose to get response efficiency evaluation by using a pre-trained neural network.

IV. AIRS INFERENCE PROCESS

The response efficiency is an essential factor to achieve an adaptive AIRS. This factor is used in the inference process that performs the AIRS reasoning to select the best response. Furthermore, the inference is based on metrics that are defined and analyzed in [18].

Inference process has the following steps:

1. Collecting information from intrusion when it arrives: System context, network context and reports from IDSs.
2. Inference of a recommended set of responses:
    a. If the intrusion is similar to a previous one, the previously selected response is executed if the proposed neural network algorithm indicated that this response was satisfactory.
    b. In other case, recommended responses are inferred based on polices, metrics, intrusion type and context parameters.

3. Optimal response according to the importance of the asset committed is selected. For significant and critical assets, we consider the response efficiency measured with the values given by neural network in previous executions of the response.

V. NEURAL NETWORK

Artificial Neural Networks are interconnected networks in parallel with a hierarchical organization. These Neural Networks attempt to interact with real world objects in the same ways as nervous system does [19].

Neurons are interconnected by their synapses, allowing transmission of information. The connections are not all equal, so you have to assign weights to the connections, ($W_{i,j,k}$).

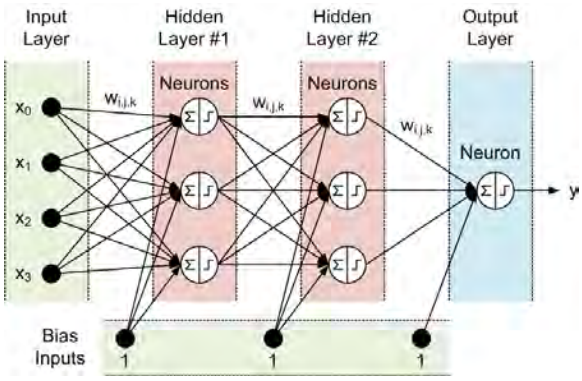

Fig. 2. Multi-layer topology of Artificial Neural Networks

The weights obtained after the learning phase determine the network output so they become the memory of neural networks.

Neural networks are particularly useful to solve problems that cannot be expressed as step by step problems, such as pattern recognition and classification. In our case, it will be used for intrusions response classification.

In particular, a Backpropagation algorithm is used to evaluate the response selected by the AIRS. This algorithm solves our problem because it determines the satisfaction of the response for any system or intrusion.

Moreover, we propose the use of a Backpropagation algorithm, as it is possible to have a training set. Supervised learning provides the following benefits:

- The network converges faster than using an unsupervised algorithm.
- Learning is based on previous observations collected by a set of instances classified during experimentation on a test system.

A trained neural network takes input samples and sorts them into groups. These can be fuzzy, i.e. the boundaries are not clearly defined, or with defined borders if thresholds are selected. The proposed system is evaluating the response with real values and not using thresholds, so that after calculating the total satisfaction of the response, the AIRS makes the decision of whether or not the response is good enough.

A. *Topology*

The generic topology of a multi-layer neural network is represented in Fig. 2. Neural Networks always have an input layer and an output layer with one or more neurons per layer. In this figure, the input layer has five neurons, taking into account the bias neuron and the output layer has one neuron. The subsection B explains the input layer for our goal. The output layer has one neuron to represent response efficiency. Furthermore, the number of hidden layers and neurons should be chosen considering the total number of neurons, the generalization error and the overfitting, as explained below.

The relationship between number of parameters and patterns must be within 10% or 20% so that we need enough patterns for generalization. Therefore, we must take into account this relationship to select the number of neurons in the hidden layer (or several hidden layers):

- Few neurons in the hidden layer will lead to higher training and generalization error due to underfitting.
- In contrast, if there are many neurons in the hidden layer then a low training error is obtained, but it has a high generalization error due to overfitting.

The number of hidden layers should be taken in account, since increasing the layers number can greatly increase the number of parameters. In addition, most problems can be solved optimally with one or two hidden layers.

Thus normally, the optimal neurons number in the hidden layer is 2/3 of total neurons corresponding to sum of neurons in both input and output layers.

The implementation of a Neural Network is parameterized in numbers of neurons and layers to find the best topology. This analysis will be realized in an initial training phase.

B. *Input parameters*

The input parameters could be the system and network context, but “normal context” depends on the device or system. Therefore, the selected input parameters correspond to the anomaly degree of system and network context. This allows the algorithm to be independent and it can be installed on all hosts. Anomaly degree corresponds to system and network context after executing a response compared to “normal context”. Context Anomaly Degree is calculated by entropy variance based on Shannon’s Information Theory.

The context parameters taken into account in our algorithm are: Status, latency, CPU usage, disk space, number of active processes, number of users, number of zombie processes and network assessment.

It is necessary to normalize the results of entropy variance calculated for use it in the Backpropagation algorithm, since very high values may impair algorithm effectiveness.

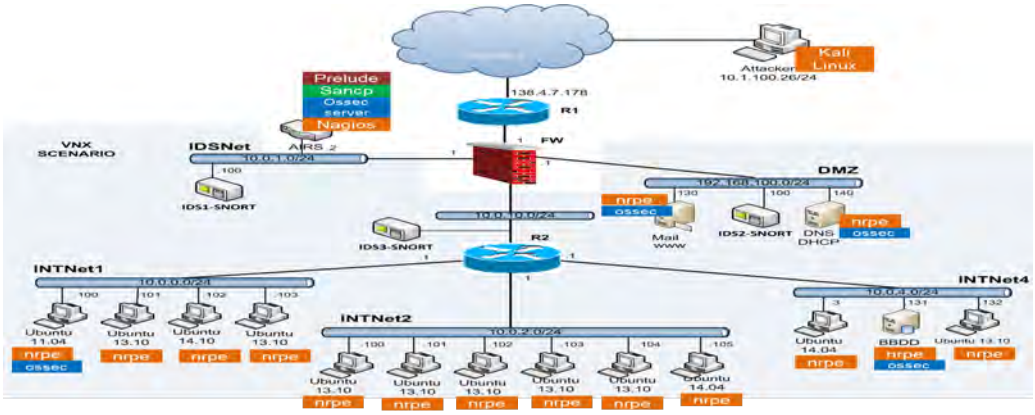

Fig. 3. Virtual test scenario

C. *Transfer function*

A sigmoid function will be used as activation function and thus to determine the level of success of the inferred response. Specifically, we use bipolar functions to achieve faster stabilization error.

We will prove two sigmoid functions:

- Bipolar logistic sigmoid function:

$$f(x) = \frac{2}{1 + e^{-x}} - 1 \tag{3}$$

- Hyperbolic tangent function:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{4}$$

D. *Stop condition*

Stop condition is determined from Mean Square Error (MSE):

$$MSE = \sum (y_{obj} - y_{en})^2 \tag{5}$$

Where y_{obj} is the desired output and y_{en} is the generated output by the algorithm.

Thus, the Backpropagation algorithm tries to find the neuron weights that minimize MSE value, in order to obtain the most accurate classification.

E. *Validation*

Validation is the last step and it is very important because it determines whether the algorithm requires additional training. To validate the generalization of neural network is necessary to have a test set.

If the training set has too much information then the neural network can suffer overlearning. Therefore, we use cross-validation method to avoid overfitting. That is, available sample is divided into two sets, training and test, with examples of all pattern types.

VI. Learning Algorithm

Backpropagation algorithm is based on gradient descent method to approach the Mean Square Error.

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij} \tag{6}$$

$$\Delta w_{ij} = \alpha \cdot \delta_j \cdot x_i \tag{7}$$

Where δ_j is the propagated error, x_i is the input value and α is the learning factor.

However, the error function usually has many local minima. Synaptic weights may depend on the mean gradient of the environment points, rather than relying on a single point, to avoid getting trapped on a local minimum. But this modification requires a large computational effort and is not efficient. Therefore, we have implemented two improvements of the algorithm to prevent local minimum: adaptive learning factor and backpropagation with momentum [20].

VII. Intrusion Response Evaluation Module

The trained neural network evaluates the response triggered by the AIRS. A satisfactory response is represented by 1, and an unsatisfactory response with a value of -1. The output in real value is taken to give more detail to the result instead of using a step function. Thus, the improvement achieved by the response against the intrusion is shown in an interval [1, -1] and it's named *SuccessLevel*. After that, the response efficiency is calculated as follows:

$$SuccessFactor = \sum_{i=0}^{j-1} SuccessLevel_i \tag{8}$$

$$ResponseEfficiency = \frac{SuccessFactor}{ExecutionTimes} \tag{9}$$

Executiontimes and j are the times that this response was triggered.

VIII. VALIDATION

Neural Networks need a training phase to construct network topology and select the best values for weights of the connections. For facilitate this task, the implementation has been made in a parameterized form. Moreover, this implementation form allows studies of efficiency of distinct architectures of Neural Network. Thus we can obtain the best result for the partial response efficiency in real time.

The validation is taken place by integrating this module in the AIRS. Then, the system will be executed in a controlled virtual environment. The virtual scenario is simulating a small organization with different servers, firewall and hosts (Fig. 3.). This scenario has been constructed using the VNX tool [21].

To collect the necessary samples of system and network context, the validation prototype uses the Nagios and Sancp tools to monitor system parameters and network traffic. Also, we will attack servers or hosts of the virtual scenario to create real alarms in the IDSs using Kali Linux and attack scripts. For example, we attack a virtual web server with DoS (Denegation of Service) using slowloris script from the attacker host [22].

In our case, we need a minimum of about 200 samples of context to train the neural network with one layer. For it, we will implement an automatic script to execute different attacks.

Once we have all the patterns, they will be presented to different topologies of the neural network. The best topology will be selected for use in the calculation of the effectiveness of the response.

This neural network will be represented as several mathematical functions based in backpropagation algorithm and the calculated weights. *SucessFactor* will be the output of this neural network when the AIRS launches a response. Finally, the response efficiency will be calculated as we explain in subsection VII.

IX. CONCLUSION AND FUTURE WORK

In this paper we propose to use Automatic Learning to train an Automated Intrusion Response System. Specifically, we have chosen Backpropagation Algorithm to measure response efficiency and thus, to provide adaptability to the AIRS.

This technology classifies any type of response, regardless of the type of intrusion detected by the system. That is, the system can obtain good results even to unknown intrusions because we have made the algorithm independent of the intrusion type.

First, it is necessary obtain patterns of system and network context for studying intrusion types. Last, we will analyze different topologies using cross-validation method with the training and test sets. Once the best neural network is selected, the system is prepared to evaluate efficiency of the triggered response for the AIRS.

We propose to use other improvements of the backpropagation learning algorithm as future work:

- Randomly initializing weights based on range.
- Using SAB method that combines adaptive learning factor and backpropagation with momentum.
- Using others bio-inspired algorithms as immune system.

ACKNOWLEDGMENT

This work has been partially funded with support from 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)

REFERENCES

[1] Symantec Corp., “Internet Security Threat Report, Vol. 17,” Abril 2012.
[2] Anderson, James. “Computer Security Threat Monitoring and Surveillance”. Washing, PA, James P. Anderson Co. 1980.
[3] H. J. Mattord. “Principles of Information Security Course Technology”. 2008. ISBN 9781423901778: 290-301.
[4] Ali Aydin M, Halim Zaim A, Gökhan Ceylan K. “A hybrid intrusion detection system design for computer network security”. Comput Elect Eng, 2009; 35(3):517–26.
[5] Stakhanova N, Basu S, Wong J. “A taxonomy of intrusion response system.” Int J Inform Comput Secur. 2007; 1(1/2):169–84
[6] J. E. López de Vergara, E. Vázquez, A. Martin, S. Dubus, M. N. Lepareux. “ Use of ontologies for the definition of alerts and policies in a network security platform”, Journal of Networks, Vol. 4, Issue 8 (2009) pp. 720-733
[7] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, “SWRL: A semantic web rule language combining OWL and RuleML”. W3C Member Submission, 21, 2004.
[8] H. Debar, D. Curry, B. Feinstein. “The Intrusion Detection Message Exchange Format (IDMEF)”. IETF Request for Comments 4765, 2007
[9] J.E. López de Vergara, V.A.Villagrá, J.I. Asensio, J. Berrocal. “Ontology-based network management: study cases and lessons learned”. J Network Syst Manage 2009; 17(3):234–54.
[10] D. L. McGuinness, F. van Harmelen. “OWL Web Ontology Language Overview. “ W3C Recommendation 2004
[11] Hu, Xunlei Rose, and Eric Atwell. "A survey of machine learning approaches to analysis of large corpora." Proceedings of the Workshop on Shallow Processing of Large Corpora, Lancaster University, UK. 2003.
[12] Ling-xi Peng, Dong-qing Xie, Ying Gao, Wen-bin Chen, Fu-fang Li, Wu Wen. “An Inmune-inspired Adaptative Automated Intrusion Response System. International Journal of Computational Intelligence” Systems, 2012, 5:5, 808-815
[13] Alireza Shameli-Sendi, Julien Desfossez, Michel Dagenais, Masoume Jabbarifar. “A Retroactive- Burst Framework for Automated Intrusion Response System”, Journal of Computer Networks and Communications, Volume 2013.
[14] Justina, Aderonke, and Adesina Simon. “A credible cost-sensitive model for intrusion response selection.” In Computational Aspects of Social Networks (CASoN), 2012 Fourth International Conference on, pp. 222-227. IEEE, 2012.
[15] Z. Mahmood, C. Agrawal, S. S. Hasan, S. Zenab, “Intrusion Detection in Cloud Computing environment using Neural Network”. International Journal of Research in Computer Engineering & Electronics, 2012. 1(1), 19-22.
[16] [20] R. S. Naoum, Abid, N. A., Z. N. Al-Sultani, “An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System”. IJCSNS, 2012. 12(3), 11.
[17] V. Mateos, V.A. Villagrá, F. Romero. “Ontologies-Based Automated Intrusion Response System. “ Computacional Intelligence in Security for information Systems 2010. Volume 85/2010. 2010:99/106

[18] V. Mateos. V. A. Villagrá, F. Romero, J. Berrocal. “Definition of response metrics for an ontology-based Automated Intrusion Response Systems.” Computers & Electrical Engineering. 2012.
[19] J. Heaton. “Introduction to Neural Networks for Java”. 2nd Edition.
[20] Rumelhart, D.E., Hinton, G.E. y Williams, R.J. (1986). Learning internal representations by error propagation. En: D.E. Rumelhart y J.L. McClelland (Eds.). Parallel distributed processing (pp. 318-362). Cambridge, MA: MIT Press
[21] Galán F, Fernández D, López de Vergara JE, Casellas R. Using a model-driven architecture for technology-independent scenario configuration in networking testbeds. IEEE Commun Mag 2010:132–141.
[22] KATKAR, Vijay, et al. Detection of DoS/DDoS Attack against HTTP Servers Using Naive Bayesian. En Computing Communication Control and Automation (ICCUBEA), 2015 International Conference on. IEEE, 2015. p. 280-285.