A NEURAL-VISUALIZATION IDS FOR HONEYNET DATA

ÁLVARO HERRERO[1]
Department of Civil Engineering, University of Burgos, Burgos (Spain)

URKO ZURUTUZA[2]
Electronics and Computing Department, Mondragon University, Arrasate-Mondragon (Spain)

EMILIO CORCHADO[3]
Departamento de Informática y Automática, Universidad de Salamanca, Salamanca (Spain)
Visiting Professor at VŠB-Technical University of Ostrava,Ostrava, Czech Republic

This study presents a novel neural intelligent system that provides network managers with a visualization of the monitored network, in order to reduce the widely known high false-positive rate associated to misuse-based Intrusion Detection Systems (IDSs). Differentiating from previous work, the proposed unsupervised neural system generates an intuitive visualization of the captured traffic rather than network statistics, letting security personnel understand what is going on in the monitored network. It is based on the use of different neural projection and unsupervised methods for the visual inspection of honeypot data, and may be seen as a complementary network security tool that sheds light on internal data structures through visual inspection of the traffic itself. Furthermore, it is intended to check and understand the performance of Snort (a well-known misuse-based IDS) through the visualization of attack patterns. Empirical verification and comparison of the proposed projection methods are performed in a real domain where two different case studies are defined and analyzed.

Keywords: Artificial Neural Networks, Unsupervised Learning, Projection Models, Network & Computer Security, Intrusion Detection, Honeypots.

1 Introduction

A network attack or intrusion will inevitably violate one of the three computer security principles -availability, integrity and confidentiality- by exploiting certain vulnerabilities such as Denial of Service (DoS), Modification and Destruction[1]. One of the most harmful issues of attacks and intrusions, which increases the difficulty of protecting computer systems, is precisely the ever-changing nature of attack technologies and strategies.

For that reason alone, among others, IDSs[2, 3, 4, 5, 6] have become an essential asset in addition to the computer security infrastructure of most organizations. In the context of computer networks, an IDS can roughly be defined as a tool designed to detect suspicious patterns that may be related to a network or system attack. Intrusion Detection (ID) is therefore a field that focuses on the identification of attempted or ongoing attacks on a computer system (Host IDS - HIDS) or network (Network IDS - NIDS).

Visual inspection of traffic patterns is an alternative and crucial aspect in network monitoring[7]. Visualization is a critical issue in the computer network defense environment, which serves to generate a synthetic and intuitive representation of the current situation for the network manager. As a result, several research initiatives have recently applied information visualization to this challenging task[8, 9, 10, 11]. Visualization techniques typically aim to make the available statistics supplied by traffic-monitoring systems more understandable in an interactive way. They therefore focus on traffic data as well as on network topology. Regardless of their specific characteristics, these methods all map high-dimensional feature data into a low-dimensional space for presentation purposes. The baseline of the novel research presented in this study is that Artificial Neural

Networks (ANNs)[12, 13, 14, 15, 16, 17], in general, and unsupervised connectionist models, in particular, can prove quite adequate for the purpose of network data visualization through dimensionality reduction[18, 19, 20]. As a result, unsupervised projection models[21, 22] are applied in the present research for the visualization and subsequent analysis of network traffic data collected by a network of honeypots, also known as a honeynet.

A honeypot has no authorized function or productive value within the corporate network other than to be explored, attacked or compromised[23]. A honeypot should not receive any traffic at all. As a consequence, all the traffic arriving at any honeynet sensor must be considered as suspicious by default. Thus every packet should be considered as an attack or at least as a piece of a multi-step attack. Recent work[24, 25] show that indeed not every packet is a part of an attack, but most of the traffic corresponds to infected computers, while the rest come from misconfigured devices. Numerous studies propose the use of honeypots to detect automatic large scale attacks; honeyd[26] and nepenthes[27] among others. The first Internet traffic monitors known as Network Telescopes, Black Holes or Internet Sinks were presented by Moore *et al*.[28] This paper advances previous preliminary work[29, 30, 31] on visual analysis of Honeynet data by the same authors, as it is described in the conclusions section.

The remaining five sections of this study are structured as follows: section 2 briefly describes the topic of computer and network security (mainly Intrusion Detection). Section 3 presents the novel approach proposed for ID while the neural projection and visualization techniques applied in this research are described in section 4. Some experimental results for two different real-life datasets are then presented and comprehensively described in section 5. Finally, the conclusions of this interdisciplinary study and the future research lines are discussed in section 6.

2 Computer and Network Security

This section introduces the main concepts of computer and network security that are the foundations of this novel study.

2.1 Intrusion Detection Systems

Intrusions can be produced by attackers that access the system, by authorized users that attempt to obtain unauthorized privileges, or by authorized users that misuse the privileges given to them. The complexity of such situations increases in the case of distributed network-based systems and insecure networks. When attackers try to access a system through external networks such as the Internet, one or several hosts may be involved. From a victim's perspective, intrusions are characterized by their manifestations, which might or might not include damage[32]. Some attacks may produce no manifestations while some apparent manifestations can be produced by system or network malfunctions.

An IDS can be defined as a piece of software that runs on a host, which monitors the activities of users and programs on the same host and/or the traffic on networks to which that host is connected[33]. The main purpose of an IDS is to alert the system administrator to any suspicious and possibly intrusive event taking place in the system that is being analyzed. Thus, they are designed to monitor and to analyze computer and/or network events in order to detect suspect patterns that may relate to a system or network intrusion.

Ever since the first studies in this field in the 80s[3, 34], the accurate, real-time detection of computer and network system intrusions has always been an interesting and intriguing problem for system administrators and information security researchers. It may be attributed on the whole to the dynamic nature of systems and networks, the creativity of attackers, the wide range of computer hardware and operating systems and so on. Such complexity arises when dealing with distributed network-based systems and insecure networks such as the Internet[35].

A standard characterization of IDSs, based on their detection method, or model of intrusions, defines the following paradigms:

- **Anomaly-based ID** (also known as behaviour-based ID): the IDS detects intrusions by looking for activity that differs from the previously defined “normal” behaviour of users and/or systems. In keeping with this idea, the observed activity is compared against "predefined" profiles of expected normal usage. It is assumed that all intrusive activities are necessarily anomalous. In real-life environments, instead of their being identical, the set of intrusive activities only intersects the set of anomalous activities in some cases. As a consequence[36], anomalous activities that are not intrusive are flagged as intrusive (i.e. false positives) and intrusive activities that are not anomalous are not flagged up (i.e. false negatives).

Anomaly-based IDSs can support detection of novel (zero-day) attack strategies but may suffer from a relatively high rate of false positives[37],

- **Misuse-based ID** (also known as knowledge-based ID): intrusions are detected by checking activity that corresponds to known intrusion techniques (signatures) or system vulnerabilities. Misuse-based IDSs are therefore commonly known as signature-based IDSs. They detect intrusions by exploiting the available knowledge on specific attacks and vulnerabilities. As opposed to anomaly detection, misuse detection assumes that each intrusive activity can be represented by a unique pattern or signature[38]. This approach entails one main problem; intrusions whose signatures are not archived by the system can not be detected. As a consequence, a misuse-based IDS will never detect a 0-day attack[38]. The completeness of such IDSs requires regular updating of their knowledge of attacks.
- **Specification-based ID**: it relies on program behavioural specifications reflecting system policies that are used as a basis to detect attacks[39].

2.1.1 Snort

Snort, a libpcap-based[40] lightweight network intrusion detection system, is one of the most widely deployed IDSs. It is a network-based, misuse-based IDS. Snort detects many types of malicious activity in the packet payload that can be characterized in a unique detection signature. It is focused on collecting packets as quickly as possible and processing them in the Snort detection engine. It is composed of three primary modules: a packet decoder, a detection engine and a logging and alerting subsystem.

Even if the capabilities of Snort allow a deep analysis of the traffic flows, what is of interest in this research is the detection, alerting and logging of the network packets as they arrive to the Honeynet system. Snort is used as a network data classifier, without discarding any packet. In that sense, in addition to the default rules of the Snort community, three basic rules that log all TCP, UDP and ICMP traffic are included, as shown in Table 1.

Table 1. Snort rules to log all TCP, UDP and ICMP traffic.

alert tcp $EXTERNAL_NET any ->$HOME_NET any (msg:"TCP"; sid:1000001;)
alert udp $EXTERNAL_NET any ->$HOME_NET any (msg:"UDP"; sid:1000002;)
alert icmp $EXTERNAL_NET any ->$HOME_NET any (msg:"ICMP"; sid:1000003;)

On the other hand, each incoming packet is inspected and compared with the default rule base. This way, besides alerting when the packet matches the three signatures shown above, many of them also match the Snort rule-base signatures. Thereby, even if a big amount of packets cause more than one alarm to be triggered, it facilitates a simple way to separate the alarm set into two subsets:

- Alarms that have been triggered when matching the Snort default rule base. This dataset can be considered as known attack data.
- Alarms that did not match any of the known attack rules. Considered as the unknown data.

These two subsets will allow network administrators to distinguish between the known and unknown traffic. This permits testing the success rate of Snort, and also visualizing the unknown traffic looking for new and unknown attacks. A clear advantage of using Snort IDS on this study is its ease of use, configuration and development of new rules.

2.2 Honeypots and Honeynets

In the last few years, two monitoring systems for automatic large-scale attack detection have been proposed: honeypots and network telescopes. Since 1992, honeypots are used to deceive attackers to learn from the new attacks they accomplish[41, 42, 43, 44]. A honeypot is a decoy system consisting of some vulnerable computing resource used to distract attackers, as an early warning system for new attack proliferations and to ease the later analysis of the attacks (forensics)[45]. When used to monitor activities derived from automatic attacks based on random or pseudo-random scanning, these systems have certain particularities. Unassigned IP addresses are given to these honeypots so every time a honeypot receives a connection request, it will be considered as suspicious. Nevertheless, the interaction level of the honeypot is fundamental. The higher the interaction between the honeypot and the attacker (response to TCP connection request for example), the more information can be gathered and therefore a higher knowledge about the attack will be obtained. A system with a low level of

interaction will also be valid to analyze the noise level, detect infected hosts, etc.

One of the most extended classifications of honeypots takes into account their level of interaction. Low interaction honeypots offer limited interaction with attackers and the most common ones only simulate services and operating systems. High interaction honeypots follow a different strategy: instead of using simulated services and operating systems, real systems and applications are used, usually running in virtual machines.

Somewhere between the two ones are medium interaction honeypots, which also emulate vulnerable services, but leave the operating system to manage the connections with their network protocol stack. Recently, a new type of honeypot has been proposed as a response to the behavioural change observed in the attackers. Instead of waiting for the attackers to reach traditional honeypots, client side honeypots, also known as honeyclients, scan communication channels looking for malware.

This study, based on the analyzed case studies, is focused on medium interaction honeypots.

Different platforms exist as observatories of malicious threats using honeypots. Examples are NoAH[26] and SGNET[27]. In this research, we are following this approach, based on the application of unsupervised learning to network level packets collected from a honeypot-based observatory.

3 A Neural Visualization-based Approach for Data Monitored by Honeypots

This study proposes the application of neural projection models for the visualization of network traffic obtained by honeypots. There exist different approaches to collect attacks on the Internet, but there is still a lack of techniques that ease comprehension and analysis of the information gathered. Visualization techniques have been applied to massive datasets for many years. These techniques are considered a viable approach to information seeking, as humans are able to recognize different features and to detect anomalies by inspecting graphs[46]. The underlying operational assumption of the proposed approach is mainly grounded in the ability to render the high-dimensional traffic data in a consistent yet low-dimensional representation. So, security visualization tools have to map high-dimensional feature data into a low-dimensional space for presentation. One of the main assumptions of the research presented in this study is that neural projection models will prove themselves to be satisfactory for the purpose of honeynet data visualization through dimensionality reduction, analyzing complex high-dimensional datasets obtained by honeypots.

Projection methods project high-dimensional data points onto a lower dimensional space in order to identify "interesting" directions in terms of any specific index or projection. Having identified the most interesting projections, the data are then projected onto a lower dimensional subspace plotted in two or three dimensions, which makes it possible to examine the structure with the naked eye. Projection methods can be seen as smart compression tools that map raw, high-dimensional data onto two or three dimensional spaces for subsequent graphical display. By doing so, the structure that is identified through a multivariable dataset may be visually analyzed with greater ease by a naked eye.

Visualization tools can therefore support security tasks in the following way:

- Visualization tools may be understood intuitively (even by inexperienced staff) and require less configuration time than more conventional tools.
- Providing an intuitive visualization of data allows inexperienced security staff to learn more about standard network behaviour, which is a key issue in ID[47]. The monitoring task can be then assigned to less experienced security staff.
- As stated in [8] "*visualizations that depict patterns in massive amounts of data, and methods for interacting with those visualizations can help analysts prepare for unforeseen events*". Hence, such tools can also be used in security training.
- They can work in unison with some other security tools in a complementary way.

As with other machine learning paradigms, an interesting facet of ANN learning is not just that the input patterns may be precisely learned/classified/ identified, but that this learning can be generalized. Whereas learning takes place within a set of training patterns, an important property of the learning process is that the network can generalize its results on a set of test patterns that were not previously learnt. Also, their capability to identify unknown patterns fits the 0-day attack[37] detection.

Due to the aforementioned reasons, the present study approaches the analysis of attack data from a

visualization standpoint. That is, some unsupervised neural projection techniques are applied for the visualization of data monitored by honeypots. To reduce the depiction time, the continuous honeynet dataflow can be split into variable-length segments, as previously proposed[20].

3.1 Previous Work

Great effort has been devoted to the ID field up to now, but several issues concerning IDS design, development, and performance are still open for further research.

Scant attention has been given to visualization in the ID field[48], although visual presentations do help operators, in general, and security managers, in particular, to interpret large quantities of data. Most IDSs do not provide any way of viewing information other than through lists, aggregates, or trends of raw data. They can generate different alarms when an anomalous situation is detected, broaden monitoring tasks, and increase situational awareness. However, they can neither provide a general overview of what is happening in the network nor support a detailed packet-level inspection[9] as is the case under honeypots and honeynets.

Some other authors have previously addressed the analysis of traffic data and intrusion detection under the application of ANN[49, 50] and more in particular projection methods[18, 51].

The underlying idea in this research is not only to detect anomalous situations under data sets monitored by honeypots but also to visualize protocol interactions and traffic volume. Packet-based ID, that is actually performed in this present research, has several advantages[52].

Some Exploratory Projection Pursuit (EPP)[53] models have been previously applied to the ID field as part of a hybrid intelligent IDS[18, 19, 20]. Differentiating from previous studies, neural EPP models, are applied in the present study as a complementary tool to IDSs for the first time to analyze real complex high-dimensional honeynet data sets. In this sense, now the output of both the neural model and Snort are combined, together with some other customized visualizations for comprehensive analysis and understanding of network status.

4 Neural Visualization Techniques

The different projection models applied in this study are described in the following subsections.

4.1 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical model, introduced in [54] and independently in [55], that describes the variation in a set of multivariate data in terms of a set of uncorrelated variables, each of which is a linear combination of the original variables.

Its goal is to derive new variables, in decreasing order of importance, that are linear combinations of the original variables and are uncorrelated with each other.

4.2 Cooperative Maximum Likelihood Hebbian Learning

Exploratory Projection Pursuit (EPP)[53] is a more recent statistical method aimed at solving the difficult problem of identifying structure in high dimensional data. It does this by projecting the data onto a low dimensional subspace in which we search for data's structure by eye. However, not all projections will reveal this structure equally well. It therefore defines an index that measures how "interesting" a given projection is, and then represents the data in terms of projections maximizing that index.

The first step for EPP is to define which indexes represent interesting directions. "Interestingness" is usually defined with respect to the fact that most projections of high-dimensional data give almost Gaussian distributions[56]. Thus, in order to identify "interesting" features in data, it is appropriate to look for those directions onto which the data-projections are as far from the Gaussian as possible.

Two simple measures of deviation from a Gaussian distribution are based on the higher order moments of the distribution. Skewness is based on the normalized third moment and measures the deviation of the distribution from bilateral symmetry. Kurtosis is based on the normalized fourth moment and measures the heaviness of the tails of a distribution. A bimodal distribution will often have a negative kurtosis and therefore negative kurtosis would signal that a particular distribution shows evidence of clustering.

Because a Gaussian distribution with mean a and variance x is equally interesting than a Gaussian distribution with mean b and variance y - indeed this

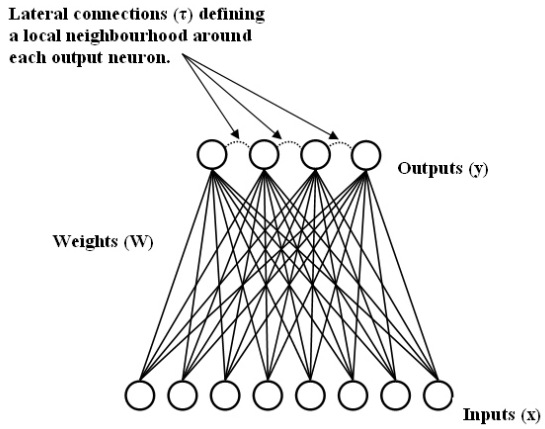

Fig. 1. CMLHL: lateral connections between neighbouring output neurons.

second order structure can obscure higher order and more interesting structure - then such information is removed from the data ("sphering").

Cooperative Maximum Likelihood Hebbian Learning (CMLHL)[21, 57] is based on Maximum Likelihood Hebbian Learning (MLHL)[21, 58], an EPP connectionist model. CMLHL includes lateral connections [57, 59] derived from the Rectified Gaussian Distribution (RGD) [60]. The RGD is a modification of the standard Gaussian distribution in which the variables are constrained to be non-negative, enabling the use of non-convex energy functions. The CMLHL architecture is depicted in Fig. 1, where lateral connections are highlighted.

Lateral connections used by CMLHL are based on the mode of the cooperative distribution that is closely spaced along a non-linear continuous manifold. Due to this, the resultant net can find the independent factors of a dataset in a way that captures some type of global ordering.

Considering an N-dimensional input vector (x), an M-dimensional output vector (y) and with W_{ij} being the weight (linking input j to output i), CMLHL can be expressed as:

Feed-forward step:

$$y_i = \sum_{j=1}^{\mathbf{N}} W_{ij} x_j, \forall i \qquad (1)$$

Lateral activation passing:

$$y_i(t+1) = \left[y_i(t) + \tau\left(b - Ay\right) \right]^+ \qquad (2)$$

Feedback step:

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i, \forall j \qquad (3)$$

Weight change:

$$\Delta W_{ij} = \eta . y_i . sign\left(e_j\right) | e_j |^p \qquad (4)$$

Where: η is the learning rate, τ is the "strength" of the lateral connections, b the bias parameter and p is a parameter related to the energy function [21, 57, 58].

A is a symmetric matrix used to modify the response to the data whose effect is based on the relation between the distances among the output neurons. It is based on the Cooperative Distribution, but to speed learning up, it can be simplified to:

$$A(i, j) = \delta_{ij} - \cos\left(2\pi\left(i - j\right)/ M\right) \qquad (5)$$

Where δ_{ij} is the Kronecker delta.

CMLHL has already proved to successfully perform data visualization. It was initially applied to the artificial vision field[57, 59] and then to some other problems[61, 62, 63].

4.3 Curvilinear Component Analysis

Curvilinear Component Analysis (CCA)[56] is a nonlinear dimensionality reduction method. Developed as an improvement on the SOM, it tries to circumvent the limitations inherent in previous linear models such as PCA.

The principle of CCA is a self-organized neural network performing two tasks: a vector quantization of the submanifold in the data set (input space) and a nonlinear projection of these quantising vectors toward an output space, providing a revealing view of the way in which the submanifold unfolds. Quantization and nonlinear mapping are separately performed by two layers of connections: firstly, the input vectors are forced to become prototypes of the distribution using a vector quantization (VQ) method; then, the output layer builds a nonlinear mapping of the input vectors by considering Euclidean distances.

In the vector quantization step, the input vectors (x_i) are forced to become prototypes of the distribution by using competitive learning and the regularization method[57] of vector quantization. Thus, this step, which is intended to reveal the submanifold of the distribution, regularly quantizes the space covered by the data, regardless of the density. Euclidean distances between these input vectors $\left(X_{ij} = d\left(x_i, x_j\right)\right)$ are considered,

as the output layer has to build a nonlinear mapping of the input vectors. The corresponding distances in the output space are also used $\left(Y_{ij} = d\left(y_i, y_j\right)\right)$.
Perfect matching is not possible at all scales when the manifold is "unfolding", so a weighting function $\left(F\left(Y_{ij}, \lambda_y\right)\right)$ is introduced, yielding the quadratic cost function:

$$E = \frac{1}{2}\sum_i \sum_{j \neq i} \left(X_{ij} - Y_{ij}\right)^2 F\left(Y_{ij}, \lambda_y\right) \qquad (6)$$

Where: **¡Error! No se pueden crear objetos modificando códigos de campo.** is a user-tuned parameter allowing an interactive selection of the scale at which the unfolding takes place.
As regards its goal, the projection part of CCA is similar to other nonlinear mapping methods, in that it minimizes a cost function based on interpoint distances in both input and output spaces. Instead of moving one of the output vectors (y_i) according to the sum of the influences of every other y_j (as would be the case for a stochastic gradient descent), CCA proposes pinning down one of the output vectors (y_i) "temporarily", and moving all the other y_j around, disregarding any interactions between them. Accordingly, the proposed "learning" rule can be expressed as:

$$\Delta y_j = \alpha(t) F\left(Y_{ij}, \lambda_y\right)\left(X_{ij} - Y_{ij}\right)\frac{y_j - y_i}{Y_{ij}} \ \forall j \neq i \quad (7)$$

Where: λ is the step size that decreases over time.

4.4 Self Organizing Map

The Self-Organizing Map (SOM)[65, 66] was developed as a visualization tool for representing high dimensional data on a low dimensional display. It is also based on the use of unsupervised learning. However, it is a topology preserving mapping model rather than a projection architecture.
Typically, the array of nodes is one or two-dimensional, with all nodes connected to the N inputs by an N-dimensional weight vector. The self-organization process is commonly implemented as an iterative on-line algorithm, although a batch version also exists. An input vector is presented to the network and a winning node, whose weight vector W_c is the closest (in terms of Euclidean distance) to the input, is chosen:

$$c = \arg\min_i\left(\left\|\mathbf{x} - W_i\right\|\right) \qquad (8)$$

Data vectors are quantized to the reference vector in the map that is closest to the input vector. The weights of the winning node and the nodes close to it are then updated to move closer to the input vector. There is also a learning rate parameter (η) that usually decreases as the training process progresses. The weight update rule for N inputs is defined as follows:

$$\Delta W_i = \eta h_{ci}\left[\mathbf{x} - W_i\right], \forall i \in N^{(c)} \qquad (9)$$

Where: W_i is the weight vector associated to the neuron *i*, **x** is the input vector, and *h* is the neighborhood function.
When this algorithm is sufficiently iterated, the map self-organizes to produce a topology-preserving mapping of the lattice of weight vectors to the input space based on the statistics of the training data.

5 Experimental Study

Researchers usually make use of well-known attack datasets such as the DARPA dataset[67, 68, 69] or the KDD Cup '99 sub-dataset[70, 71] in order to validate their developed systems. However, these data are simulated, non-validated and irregular so they are not fully reliable[72, 73]. Even if the results obtained by such systems are good, no one can assure that the applied algorithms will make the system more secure or will detect real attacks. This is the main reason of using two real traffic data sets coming from a running honeynet in this research.
The experimental work has been done by using data from real traffic that reached the Euskalert network[74]. These data are depicted through different neural projection and visualization techniques in order to discover real attack behaviour and strategies.
The Euskalert project[74] has deployed a network of honeypots in the Basque Country (northern Spain), where eight companies and institutions have installed one of the project's sensors behind the firewalls of their corporate networks. The honeypot sensor transmits all the traffic received to a database via a secure communication channel. These partners can consult information relative to their sensor as well as general statistics in the project's website. Once a big amount of data has been collected, the information available can be

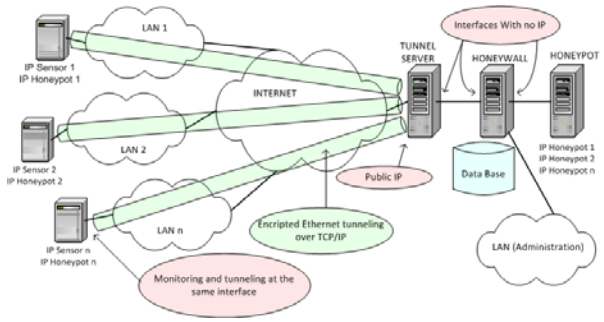

Fig. 2. Architecture of the Euskalert network.

used to analyze attacks received by the honeynet at network and application level.

Euskalert is a distributed honeypot network based on a Honeynet GenIII architecture[53]. The developed architecture of Euskalert is shown in Fig. 2. The various sensors installed in corporate networks of the different participants are shown in the left of Fig. 2.

Every sensor has a permanently established encrypted connection (using different virtual private networks) to a tunnel server. The latter is in the DMZ (Demilitarized Zone) of Mondragon University. Any attack to one of the sensors is redirected through these tunnels to reach the Honeypot (right side of Fig. 2), which is responsible for responding to any connection attempt. The traffic also passes through a server responsible for collecting all the information which is then displayed on the Web platform[74].

This honeypot system has received about 164 packets a day on average. All the incoming traffic is analyzed by the Snort IDS, and an alert is launched whenever a packet matches a known signature.

The following features were extracted from each one of the records in the dataset:

- **Time**: the time when the attack was detected. Difference in relation to the first attack in the dataset (in minutes).

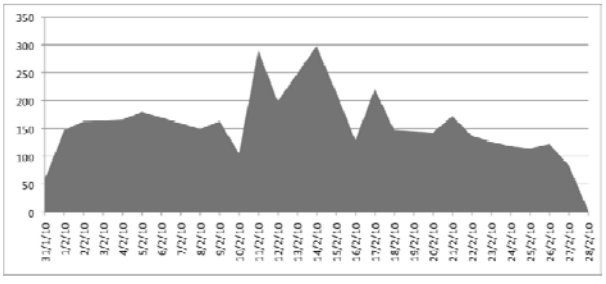

Fig. 3. Temporal distribution of the traffic volume in terms of number of packets captured by Euskalert during February 2010.

- **Protocol**: either TCP, UDP or ICMP (codified as three binary features).
- **Ip_len**: number of bytes in the packet.
- **Source Port**: number of the port from which the source host sent the packet. In the case of the ICMP protocol, this represents the ICMP type field.
- **Destination Port**: destination host port number to which the packet was sent. In the case of the ICMP protocol, this represents the ICMP type field.
- **Flags**: control bits of a TCP packet, which contains 8 1-bit values.

Two different real-life case studies are analyzed in this research as attack behavior may change in time. First, a snapshot of one-month data (February, 2010) is analyzed to observe its internal structure. A five-month period (February-June, 2010) is taken later to analyze how attacks have changed and new trends are discovered.

The previously introduced projection models have been applied to these two case studies, whose results are shown and described in the following subsections.

5.1 Case study 1: a 1-month dataset

For this real case study, the logs coming from Euskalert and Snort have been gathered during one month (February, 2010). Fig. 3 shows the traffic volume in terms of number of packets received for that period of time. The amount of daily network traffic received changes from day to day. Most of the traffic is malicious, thus it does not follow any predefined pattern or distribution, being the traffic volume unpredictable. In this case, days 11/02/2010 and 14/02/2010 can reflect a new worm outbreak, or a Denial of Service attack. Furthermore, sensors collecting traffic tend to loose connection to the server from time to time, and this also generates drops in traffic (see 10/02/2010 or 28/02/2010).

The February 2010 dataset contains a total of 3798 packets, including TCP, UDP and ICMP traffic received by the distributed honeypot sensors. The characterization of the traffic in the dataset is shown in Table 2. The table shows which alerts have been triggered in that period of time and their percentage. Those signatures starting with “Wormledge” are automatically generated and not present in the default signature database.

Table 2. Characterization of traffic data captured by Euskalert, during February, 2010.

Signature	# Packets	% of traffic
Unknown Traffic	3404	89,62
POLICY Reserved IP Space Traffic - Bogon Nets 2	127	3,34
WORM Allaple ICMP Sweep Ping Inbound	58	1,52
ICMP PING	75	1,97
Wormledge, microsoft-ds, smb directory packet (port 445)	34	0,89
Wormledge, KRPC Protocol, BitTorrent	11	0,28
Wormledge, NetBios Session Service (port 139)	7	0,18
Wormledge, NetBios Name Query (udp port 137)	7	0,18
Wormledge, Microsoft RPC Service, dce endpoint resolution (port 135)	7	0,18
WEB-IIS view source via translate header	6	0,15
SCAN LibSSH Based SSH Connection	5	0,13

From this dataset, it may be said that a misuse detection-based IDS such as Snort is only capable of identifying about 10.38% of bad-intentioned traffic. Furthermore, it was demonstrated that only 2% of the unsolicited traffic was identified by the IDS when automatically generated signatures were included from a previous work[24]. Thus, a deeper analysis of the data is needed in order to discover the internal structure of the remaining 90% of the traffic. Explaining the behaviour of the unknown traffic is a difficult task that must be performed to better protect computer networks and systems. In order to obtain more knowledge, several neural projection models have been applied and the results and conclusions obtained are shown in the following sub-sections.

In the visualizations obtained, the data are depicted with different colors and shapes, taking into account the different original features of the data. In the shown projections (Fig. 4 to Fig. 7), the axes are combinations of the features contained in the original datasets. Then, the X and Y axes of the projections can not be associated to a unique original feature[21, 57].

5.1.1 CMLHL Projections

The CMLHL-training parameter values for the projections in this section were: number of iterations = 10,000, learning rate = 0.0208, p parameter = 2.1429, and τ parameter = 0.067.

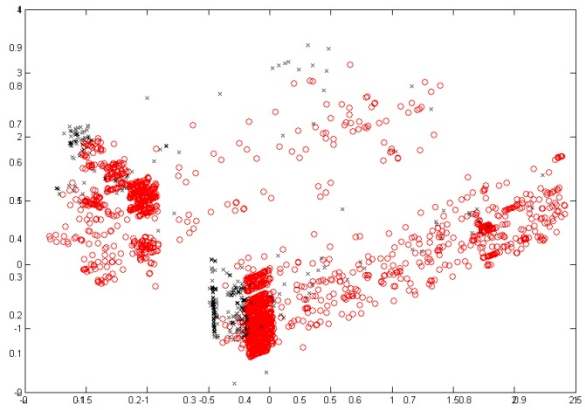

Fig. 5. CMLHL projection of 1-month data - Snort output. Parameters: number of iterations = 10,000, learning rate = 0.0208, p parameter = 2.1429, and τ parameter = 0.067.

Fig. 4 shows the CMLHL projection by considering the output generated by Snort. Packets that triggered any alarm are depicted as black crosses while packets that were identified as unknown are depicted as red circles.

After analyzing this projection (Fig. 4), it can be confirmed the poor detection performance of Snort IDS when filtering honeypot traffic. CMLHL provides a way of differentiating known from unknown traffic at a naked eye. Most of the traffic corresponds to unknown packets, or at least to traffic that Snort is not capable of identifying using all of its predefined rule sets.

Fig. 5 shows the CMLHL projection by considering the detection timestamp (in minutes) to depict the packets; from 0 to 6692: red circles, from 6693 to 13384: black crosses, from 13385 to 20076: green pluses, from 20077 to 26768: magenta stars, from 26769 to 33460: yellow squares, and from 33461 to 40148: cyan diamonds.

The traffic temporal evolution on that month shows that same traffic patterns repeat over time, as almost every

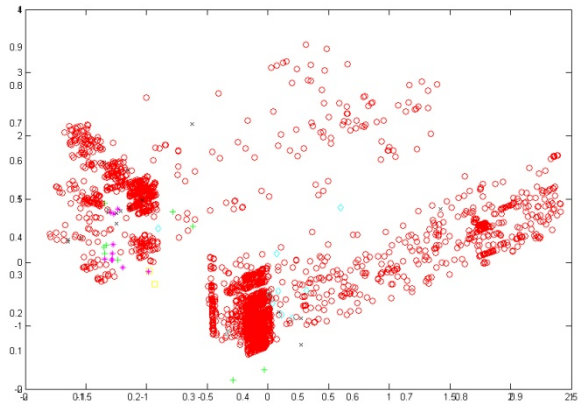

Fig. 4. CMLHL projection of 1-month data - IP length. Parameters: number of iterations = 10,000, learning rate = 0.0208, p parameter = 2.1429, and τ parameter = 0.067.

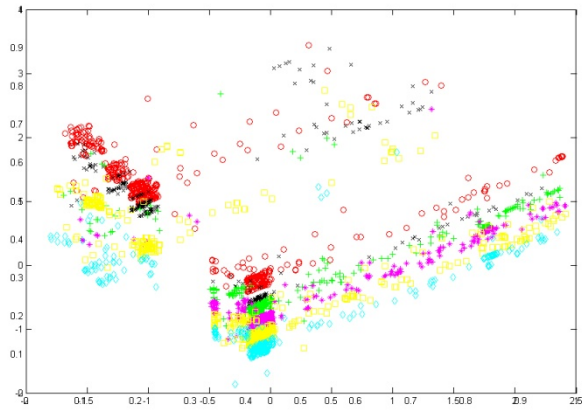

Fig. 6. CMLHL projection of 1-month data - Timestamp. Parameters: number of iterations = 10,000, learning rate = 0.0208, p parameter = 2.1429, and τ parameter = 0.067.

cluster have similar shapes. This happens with both known and unknown traffic (shown in Fig. 4). It can be concluded that anomalous or unknown behavior is not a one-off event, but a recurring pattern in time instead.

Fig. 6 shows the CMLHL projection by considering the IP length (in bits) to depict the packets; from 28 to 273: red circles, from 274 to 519: black crosses, from 520 to 765: green pluses, from 766 to 1011: magenta stars, from 1012 to 1257: yellow squares, and from 1258 to 1500: cyan diamonds.

Most of the traffic is composed of small packets, but it can also be observed that very large packets are received by the honeypot sensors. Attackers must create specially prepared packets in order to overflow listening service's memory buffers and stacks first, and execute arbitrary commands later. This payload, known as shellcode, can have a large size. These are synonyms of receiving malware, vulnerability exploits, or DoS attacks.

All those visualizations are in general very helpful information for explaining and helping security administrators to know the different traffic behavior that reach their systems.

5.1.2 Comparative Study

The obtained CMLHL projections are compared with two other dimensionality-reduction models (CCA and SOM). Several experiments were required to tune CCA to different options and parameters: initialization, epochs and distance criterion, among others. In the case of SOM, other parameters, such as grid size, batch/online training, initialization, number of iterations and distance criterion were tuned. Only the best results (from the standpoint of the projection) for each model,

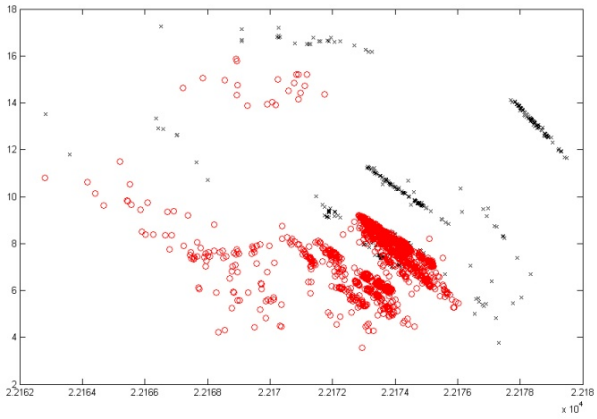

Fig. 7. CCA projection of 1-month data - Snort output. Parameters: standardized Euclidian distance, lambda = 230,000, alpha = 0.5 and 10 epochs.

which were obtained after the tuning stage, are included in this section.

5.1.2.1 Curvilinear Component Analysis

Fig. 7 shows the CCA projection of the case study 1 by using the Snort output. The following parameters were tuned: alpha, lambda, number of epochs and distance criterion. The final selected parameter values were: standardized Euclidian distance, lambda = 230,000, alpha = 0.5 and 10 epochs.

This projection (Fig. 7) shows a visual explanation of the distribution of packets identified by Snort and those which are not. CCA is much more resource demanding than the other models as the pair-wise distance matrix must be calculated.

5.1.2.2 Self-Organizing Map

Finally, the SOM was also applied to the 1-month dataset. Fig. 8 shows the SOM map of the case study 1

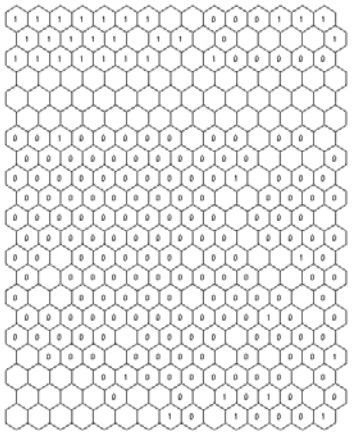

Fig. 8. SOM mapping of 1-month data - Snort output. Parameters: linear initialization, batch training, hexagonal lattice, "Cut Gaussian" neighborhood function, and grid size (determined by means of a heuristic formula) = 15x21.

by using the Snort output. In order to analyze the resulting maps, the two different Snort outputs (1= triggered alarm, 0 = no triggered alarms) are assigned to SOM neurons.

For the SOM, the following options and parameters were tuned: grid size, batch/online training, initialization, number of iterations and distance criterion among others. The used parameter values were: linear initialization, batch training, hexagonal lattice, "Cut Gaussian" neighborhood function, and grid size (determined by means of a heuristic formula) = 15x21.

After analyzing this mapping, it can be concluded that SOM is not able to cluster the data distinguishing the traffic classification of Snort (alarm/no alarm). The cluster in the upper left section (Fig. 8) is the only one identifying traffic of only one class (packets that triggered an alarm), while the other ones identify traffic from the two classes.

5.2 Case study 2: a 5-month real dataset

For this experiment, we have analyzed the logs coming from Euskalert and Snort gathered during five months starting from February, 2010. Fig. 9 shows the traffic volume in terms of number of packets received for that period of time.

The dataset contains a total of 22,601 packets, including TCP, UDP and ICMP traffic received by the distributed honeypot sensors. The characterization of the traffic in this dataset is contained in Table 3, which shows which alerts have been triggered in that period of time and their percentage. Those signatures starting with "Wormledge" are automatically generated and not present in the default Snort signature database. As Table 3 shows, the biggest group of signatures are those generated for unknown packets (both TCP, UDP and ICMP), and also the automatically generated signatures from a previous work[24].

From this dataset, it may be said that a misuse detection-based IDS such as Snort is only capable of identifying less than 3,75% (847 packets out of 22,601) of bad-intentioned traffic. Comparing to the initial month, the percentage of identified packets is smaller in this case study. There are two main reasons that might explain this fact. Firstly, there is a component of randomness, as it is not possible to know neither the volume that will generate old and new malware and traffic from misconfigured devices, nor the nature of

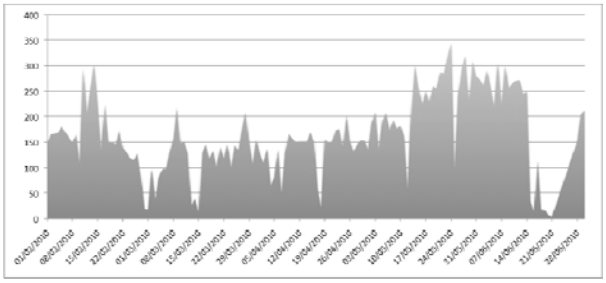

Fig. 9. Temporal distribution of the traffic volume in terms of number of packets captured by Euskalert from February to June, 2010.

that traffic (known/unknown classification by Snort IDS). Traffic bursts may happen at any time, and of any type. On the other hand, it is related with the time that passed until Snort's signatures are updated. The older the signatures are, the more likely it is to get unknown traffic. Another observation of this dataset is the higher average volume traffic found in this period. It happened because of traffic congestion, but specially due to the fact that a new sensor was added to the Euskalert platform. Thus, a deeper analysis of the data is needed in order to discover the internal structure of the remaining 96,25% of the traffic data set. As in case study 1, neural unsupervised models are applied in order to explain the behavior of the unknown traffic.

Table 3. Characterization of traffic data captured by Euskalert, from February to June, 2010.

Signature	# Packets	% of traffic
Unknown TCP packet	19096	84,49183664
Reserved IP Space Traffic - Bogon Nets	1071	4,738728375
Unknown packet	741	3,27861599
Unknown UDP packet	397	1,756559444
ICMP ping	290	1,283129065
WORM Allaple ICMP Sweep Ping Inbound	251	1,110570329
Wormledge, KRPC Protocol, BitTorrent	99	0,438033715
Wormledge, Slammer Worm	62	0,274324145
Wormledge, Microsoft-ds, smb directory packet (port 445)	62	0,274324145
Wormledge, MS-SQL-Service(port tcp 1433)	58	0,256625813
ICMP PING speedera	40	0,176983319
Wormledge, NetBios Name Query (udp port 137).	35	0,154860404
Wormledge, Possible SQL Snake/Spida Worm	34	0,150435821

Wormledge, NetBios Session Service (port 139)	34	0,150435821
SIP TCP/IP message flooding directed to SIP proxy	33	0,146011238

5.2.1 CMLHL Projections

CMLHL was applied in order to analyze the dataset described above and to identify its inner structure. The CMLHL-training parameter values for the projections in this section were: number of iterations = 30,000, learning rate = 0.01, p parameter = 1.22, and τ parameter = 0.13137.

Fig. 10 shows the CMLHL projection by considering the output generated by Snort. Packets that triggered any alarm are depicted as black crosses while packets that were not identified as anomalous are depicted as red circles.

Visualization of packets using Snort output shows the detection rate of the most used misuse-based IDS. Black crosses are the only ones detected by Snort, where all of the records constitute a suspicious activity by default. It is therefore important to use additional supporting systems, such as the visualization aids proposed in this study, to show a more comprehensive picture of what is actually happening and how an IDS is performing.

Fig. 11 shows the CMLHL projection by considering the detection timestamp to depict the packets; from 0 to 27044 minutes: red circles, from 27045 to 54089: black crosses, from 54090 to 81134: green pluses, from 81135 to 108179: magenta stars, from 108180 to 135224: yellow squares, and from 135225 to 162267: cyan diamonds.

It can be observed that groups are highly overlapped, which means that temporal distribution of attacks is very homogenous, so they constantly repeat over time. If we look at some of the attacks carefully, we still see

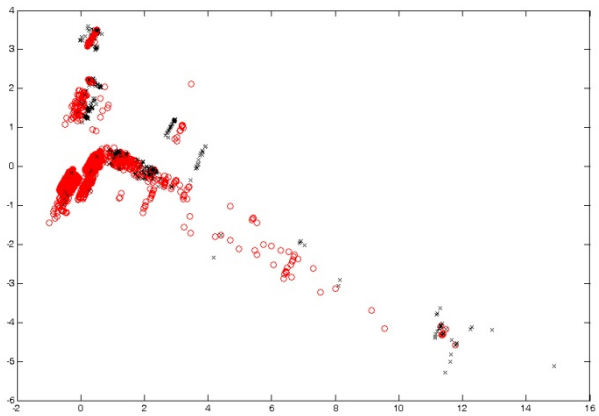

Fig. 10. CMLHL projection of 5-month data - Snort output. Parameters: number of iterations = 30,000, learning rate = 0.01, p parameter = 1.22, and τ parameter = 0.13137.

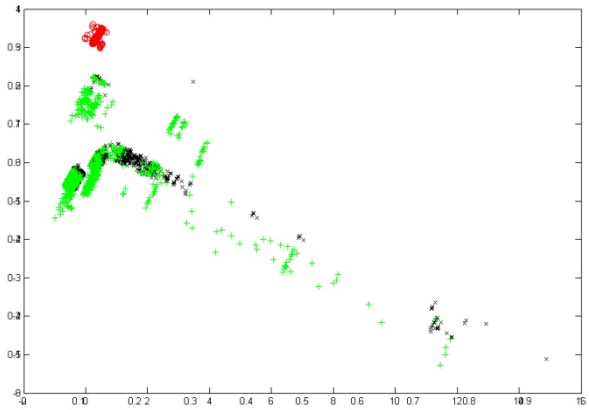

Fig. 11. CMLHL projection of 5-month data - Destination Port. Parameters: number of iterations = 30,000, learning rate = 0.01, p parameter = 1.22, and τ parameter = 0.13137.

very old worm instances such as Slammer or Blaster (see Table 3).

Fig. 12 shows the CMLHL projection by considering different ranges of the destination port to depict the packets; 3 and 8: red circles correspond to ICMP type codes, from 0 to 1023 (excluding 3 and 8): black crosses corresponding known application listening services, and from 1024 to 54612: green pluses corresponding to non-privileged ports. If we choose particular port numbers of services that are commonly exploited we would have too many clusters depicted. Choosing privileged, non-privileged and ICMP bands leverage to a clear view of the nature of the traffic observed.

This visualization shows that Euskalert receives packets and connection attempts to ports above 1023. We find two possible explanations of this observation. On the one hand, there are attack attempts to applications listening on ports above 1023. In this case we should focus on these ports and if we find any prevalence then create a new simulated service for that application. On the other hand, backscatter is received, being this port the source port of the attacker.

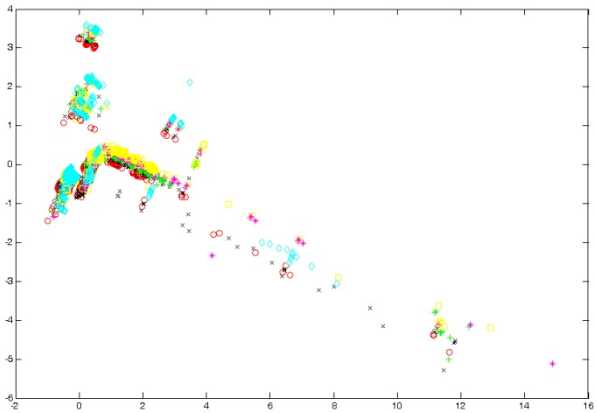

Fig. 12. CMLHL projection of 5-month data - Timestamp. Parameters: number of iterations = 30,000, learning rate = 0.01, p parameter = 1.22, and τ parameter = 0.13137.

5.2.2 Comparative Study

As in the case study 1, the CMLHL projections are compared with those of other dimensionality-reduction models; PCA and MLHL in this case.

5.2.2.1 Principal Component Analysis

Fig. 13 shows the PCA projection of the case study 2 by using the time feature to depict the packets; from 0 to 27044 minutes: red circles, from 27045 to 54089: black crosses, from 54090 to 81134: green pluses, from 81135 to 108179: magenta stars, from 108180 to 135224: yellow squares, and from 135225 to 162267: cyan diamonds. The two first principal components amount to 96.87% of original data's variance.

After analyzing this visualization it can be said that it offers a clear representation of the observed conclusions, as the packet distribution in time is similar.

5.2.2.2 Maximum Likelihood Hebbian Learning

The MLHL-training parameter values for the projection in this section were: number of iterations = 35,000, learning rate = 0.02, and *p* parameter = 0.9.

Fig. 14 shows the MLHL projection of the case study 2 by using the Snort output. This visualization shows sharply the conclusions obtained from Fig. 10. It demonstrates that most of the traffic reaching the honeypot remains unexplained, although in the MLHL projection there are no clearly defined clusters.

6 Conclusions and Future Work

Apart from the previously stated conclusions, regarding each one of the analyzed datasets, some other (more general) conclusions are provided in this section. This conclusions are different from the ones in previous studies, mainly because one of the analyzed dataset is bigger than any previous one. Thus, new attacks and situations are considered now. On the other hand, a comprehensive comparative study is now provided, compressing results from a wide range of projection/visualization models.

After comparing the different projections obtained in this study, it can be concluded that CMLHL provides a more sparse and clearer representation than the other applied projection methods. This enables the intuitive visualization of the Honeynet data, where the general structure of these data can be seen and interpreted. As an advance from previous work, the visualizations obtained through CMLHL incorporating Snort output,

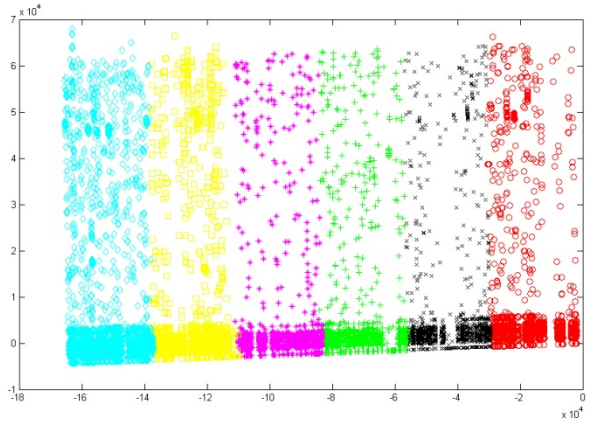

Fig. 13. PCA projection of 5-month data - Time.

give an insight of the captured honeynet data, providing useful knowledge about the attacks a network could face. The application of this neural model allows the depiction of the captured traffic itself, rather than network statistics or topology, as some previous work does.

Another contribution of this study is the fact that CMLHL is capable of analyzing big volumes of data, keeping the visualization patterns clear, thus easing the analysis of the honeypot phenomena.

It has been shown how CMLHL provides a helpful technique to visualize backscatter attacks, as well as identifying those attacks that overflow a buffer and download malware.

The proposed neural visualization advances from previous work by providing a useful insight of honeynet data to intuitively check the performance of Snort. From a general perspective, it can be seen from the Snort output visualizations how high the classification error rate of Snort is.

After getting a general idea of the dataset structure, an in-depth analysis was carried out to comprehensively

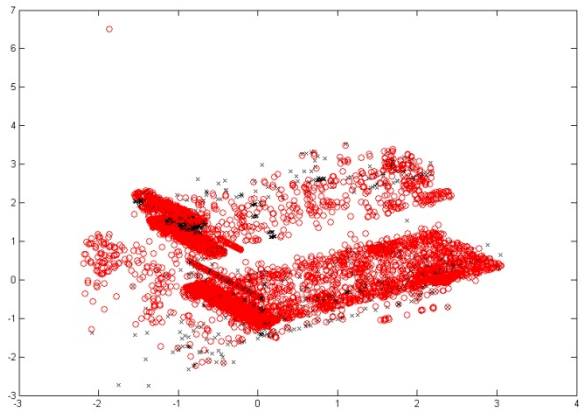

Fig. 14. MLHL projection of 5-month data - Snort output. Parameters: number of iterations = 35,000, learning rate = 0.02, and p parameter = 0.9.

analyze each one of the points in the groups identified by CMLHL (Figs 4-6 and 10-12). As a result, the following conclusions can be stated for each one of the destination ports:

- 8: We identify the type of ICMP packet by inserting its code into the field destination port. ICMP type 8 corresponds to ICMP echo or ping, used for probing the Internet, looking for victim hosts.
- 22: SSH. It seems to be a traffic flow with many packets coming from one source to one of the honeypot. They correspond to connection attempts by attackers or infected machines.
- 80: HTTP. Attackers try different vulnerabilities against web applications.
- 135: DCE endpoint resolution, used by Microsoft for Remote Procedure Call protocol. It has always been and still is one of the most exploited services by virus and worms.
- 139: NETBIOS Session Service. Plenty of attacks to this Microsoft Windows service can be found.
- 443: HTTP protocol over TLS SSL connection attempts.
- 445: SMB directly over IP. As most of the traffic in the biggest group identified by CMLHL is aimed at this destination port, we can conclude that this is a widely exploited service.
- 1433: Microsoft-SQL-Server, used by the old SQL Slammer worm.
- 1521: Oracle TNS Listener. It seems that attackers try to connect to the honeypot via Oracle service.
- 2967: Symantec System Center. Vulnerabilities have been found on Symantec service, and it is being exploited in the wild.
- 3128: Proxy Server // Reverse WWW Tunnel Backdoor, where the MyDoom worm operates.
- 3389: MS Terminal Services, used for Remote Desktop.
- 4444: This port is a common return port for the rpc dcom.c buffer overflow and for msblast rpc worm.
- 4899: Remote Administrator default port. There is a known remotely exploitable vulnerability in some radmin server versions that allows code execution.
- 5061: SIP-TLS. Used for VoIP communications.
- 5900: Virtual Network Computer or VNC, used also as a remote desktop solution.
- Port 8080: HTTP Alternate port, also used as an HTTP proxy.
- Port 19765: Used in Kademlia (Bittorrent protocol).

This deeper analysis remains necessary in order to better understand some of the visualized attacks, but CMLHL projections seem enough to obtain a fast understanding of Internet attacks.

Further work will focus on the application of different projection/visualization models as well as studying the visualization with different metrics instead of using the original features of the data.

More analysis can be done with the data, like visualization of this attack traffic by each of the honeynet infrastructure sensors. This way, one could compare the pattern of attack behavior distinguishing the Internet space placement. On the other hand, to speed up the data analysis, High Performance and Parallel Computing can be also applied.

Another interesting improvement of CMLHL visualization could be providing interactive capabilities; a user or analyst could select one or more points from the projections and the system may give details about the data behind them. In a further approach, the system could automatically generate signatures for user selected clusters, giving a solution to the big amount of Snort's undetected packets.

Enrichment of the attack dataset may also be a focus of attention. Researchers are correlating network traffic data with exploits collected during simulated vulnerability exploitation. Malware is also obtained for those attacks that aim to spread the infection. All this data needs a deeper analysis, and neural projection techniques will help in that task.

These datasets will be shared to any interested researcher upon request. This will provide a solution to a known issue, allowing other groups to compare their results.

Acknowledgements

This research has been partially supported through the Regional Government of Gipuzkoa, the Department of Research, Education and Universities of the Basque Government; and the Spanish Ministry of Science and Innovation (MICINN) under project TIN2010-21272-C02-01 and CIT-020000-2009-12 (funded by the European Regional Development Fund). This work was also supported in the framework of the IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 supported by Operational Programme 'Research and Development for Innovations' funded by Structural Funds of the European Union and state budget of the Czech Republic.

References

1. J. M. Myerson, Identifying enterprise network vulnerabilities, *International Journal of Network Management,* 12 (3) (2002) 135-144.
2. Computer security threat monitoring and surveillance, (James P. Anderson Co 1980).
3. D. E. Denning, An intrusion-detection model, *IEEE Transactions on Software Engineering,* 13 (2) (1987) 222-232.
4. T. Chih-Fong, H. Yu-Feng, L. Chia-Ying, and L. Wei-Yang, Intrusion detection by machine learning: A review, *Expert Systems with Applications,* 36 (10) (2009) 11994-12000.
5. Z. Bankovi, J. M. Moya, Á. Araujo, D. Fraga, J. C. Vallejo, and J. M. de Goyeneche, Distributed intrusion detection system for wireless sensor networks based on a reputation system coupled with kernel self-organizing maps, *Integrated Computer-Aided Engineering,* 17 (2) (2010) 87-102.
6. C. Otte and C. Störmann, Improving the accuracy of network intrusion detectors by input-dependent stacking, *Integrated Computer-Aided Engineering,* 18 (3) (2011) 291-297.
7. R. A. Becker, S. G. Eick, and A. R. Wilks, Visualizing network data, *IEEE Transactions on Visualization and Computer Graphics,* 1 (1) (1995) 16-28.
8. A. D. D'Amico, J. R. Goodall, D. R. Tesone, and J. K. Kopylec, Visual discovery in computer network defense, *IEEE Computer Graphics and Applications,* 27 (5) (2007) 20-27.
9. J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi, Focusing on context in network traffic analysis, *IEEE Computer Graphics and Applications,* 26 (2) (2006) 72-80.
10. T. Itoh, H. Takakura, A. Sawada, and K. Koyamada, Hierarchical visualization of network intrusion detection data, *IEEE Computer Graphics and Applications,* 26 (2) (2006) 40-47.
11. Y. Livnat, J. Agutter, S. Moon, R. F. Erbacher, and S. Foresti, A visualization paradigm for network intrusion detection, in *Sixth Annual IEEE SMC Information Assurance Workshop, 2005. IAW '05*,eds., 2005) pp. 92-99.
12. H. Adeli and S. L. Hung, *Machine learning - neural networks, genetic algorithms and fuzzy systems*, (John Wiley & Sons, New York, USA, 1995).
13. H. Adeli and X. Jiang, Dynamic fuzzy wavelet neural network model for structural system identification, *Journal of Structural Engineering,* 132 (2006) 102-111.
14. M. Ahmadlou and H. Adeli, Enhanced probabilistic neural network with local decision circles: A robust classifier, *Integrated Computer-Aided Engineering,* 17 (3) (2010) 197-210.
15. M. Oja, G. O. Sperber, J. Blomberg, and S. Kaski, Self-organizing map-based discovery and visualization of human endogenous retroviral sequence groups, *International Journal of Neural Systems,* 15 (3) (2005) 163-180.
16. N. Lopes and B. Ribeiro, An evaluation of multiple feed-forward networks on graphics processing units, *International Journal of Neural Systems,* 21 (1) (2010) 31-47.
17. M. L. Borrajo, B. Baruque, E. Corchado, J. Bajo, and J. M. Corchado, Hybrid neural intelligent system to predict business failure in smes, *International Journal of Neural Systems,* 21 (4) (2011) 277-296.
18. Á. Herrero, E. Corchado, P. Gastaldo, and R. Zunino, Neural projection techniques for the visual inspection of network traffic, *Neurocomputing,* 72 (16-18) (2009) 3649-3658.
19. Á. Herrero, E. Corchado, M. A. Pellicer, and A. Abraham, Movih-ids: A mobile-visualization hybrid intrusion detection system, *Neurocomputing,* 72 (13-15) (2009) 2775-2784.
20. E. Corchado and Á. Herrero, Neural visualization of network traffic data for intrusion detection, *Applied Soft Computing,* 11 (2) (2011) 2042–2056.
21. E. Corchado, D. MacDonald, and C. Fyfe, Maximum and minimum likelihood hebbian learning for exploratory projection pursuit, *Data Mining and Knowledge Discovery,* 8 (3) (2004) 203-225.
22. L. Xu, Best harmony, unified rpcl and automated model selection for unsupervised and supervised learning on gaussian mixtures, three-layer nets and me-rbf-svm models, *International Journal of Neural Systems,* 11 (1) (2001) 43-69.
23. K. A. Charles, Decoy systems: A new player in network security and computer incident response, *International Journal of Digital Evidence,* 2 (3) (2004).
24. U. Zurutuza, R. Uribeetxeberria, and D. Zamboni, A data mining approach for analysis of worm activity through automatic signature generation, in *1st ACM Workshop on AISec*,eds. (ACM, 2008) pp. 61-70.
25. F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb, Detecting network-wide and router-specific misconfigurations through data mining, *IEEE/ACM Transactions on Networking,* 17 (1) (2009) 66-79.
26. N. Provos, A virtual honeypot framework, in *13th USENIX Security Symposium*,eds., 2004) pp.
27. P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. Freiling, The nepenthes platform: An efficient approach to collect malware, in *9th International Symposium on Recent Advances in Intrusion Detection (RAID 2006)*,eds. (Springer Berlin / Heidelberg, 2006) pp. 165-184.
28. D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, Inferring internet denial-of-service activity, *ACM Transactions on Computer Systems,* 24 (2) (2006) 115-139.
29. Á. Alonso, S. Porras, E. Ezpeleta, E. Vergara, I. Arenaza, R. Uribeetxeberria, U. Zurutuza, Á. Herrero, and E. Corchado, Understanding honeypot data by an unsupervised neural visualization, in *Computational intelligence in security for information systems 2010* Springer Berlin / Heidelberg, pp. 151-160.
30. U. Zurutuza, E. Ezpeleta, Á. Herrero, and E. Corchado, Visualization of misuse-based intrusion detection: Application to honeynet data, in *6th international conference on soft computing models in industrial and environmental applications* Springer Berlin / Heidelberg, pp. 561-570.
31. Á. Alonso, S. Porras, E. Ezpeleta, E. Vergara, I. Arenaza, R. Uribeetxeberria, U. Zurutuza, Á. Herrero, and E. Corchado, On the visualization of honeypot data through projection techniques, in *10th International Conference on Mathematical Methods in Science and Engineering*,eds., 2010) pp. 1038-1049.

32. J. McHugh, A. Christie, and J. Allen, Defending yourself: The role of intrusion detection systems, *IEEE Software,* 17 (5) (2000) 42-51.
33. L. Khaled, Computer security and intrusion detection, *Crossroads,* 11 (1) (2004) 2-2.
34. J. P. Anderson, Computer security threat monitoring and surveillance, (Technical Report 1980).
35. P. G. Neumann and D. B. Parker, A summary of computer misuse techniques, in *12th National Computer Security Conference*,eds., 1989) pp. 396-407.
36. A. Sundaram, An introduction to intrusion detection, *Crossroads,* 2 (4) (1996) 3-7.
37. P. Laskov, P. Dussel, C. Schafer, and K. Rieck, Learning intrusion detection: Supervised or unsupervised?, in *13th International Conference on Image Analysis and Processing (ICIAP 2005)*,eds. F. Roli and S. Vitulano (Springer, Heidelberg, 2005) pp. 50-57.
38. J. M. Rizza, *Computer network security*, (Springer US, 2005).
39. I. Balepin, S. Maltsev, J. Rowe, and K. Levitt, Using specification-based intrusion detection for automated response, in *Sixth International Symposium on Recent Advances in Intrusion Detection (RAID 2003)*,eds. (Springer, Heiderlberg, 2003) pp. 136-154.
40. Libpcap, http://www-nrg.ee.lbl.gov/. Last access: 08/25/11
41. S. M. Bellovin, There be dragons, in *Third Usenix Security Symposium*, Citeseer, 1992), pp. 14-16.
42. B. Cheswick, An evening with berferd in which a cracker is lured, endured, and studied,eds. (Citeseer, 1990) pp. 1990.
43. Deception toolkit, http://www.all.net/dtk/. Last access:
44. L. Spitzner, *Honeypots: Tracking hackers*, (Addison-Wesley Professional, 2003).
45. L. Spitzner, Honeypots: Sticking to hackers, *Network Magazine,* 18 (4) (2003) 48-51.
46. C. Ahlberg and B. Shneiderman, Visual information seeking: Tight coupling of dynamic query filters with starfield displays, in *Readings in information visualization: Using vision to think* Morgan Kaufmann Publishers Inc., 1999), pp. 244-250.
47. J. R. Goodall, W. G. Lutters, P. Rheingans, and A. Komlodi, Preserving the big picture: Visual network traffic analysis with tnv, in *IEEE Workshop on Visualization for Computer Security (VizSEC 05)*,eds. (IEEE Computer Society, 2005) pp. 47-54.
48. J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner, State of the practice of intrusion detection technologies, (Carnegie Mellon University - Software Engineering Institute 2000).
49. T.-W. Yue and S. Chiang, A neural-network approach for visual cryptography and authorization, *International Journal of Neural Systems,* 14 (3) (2004) 175-187.
50. J. Zhou, Q. Z. Xu, W. J. Pei, Z. He, and H. Szu, Step to improve neural cryptography against flipping attacks, *International Journal of Neural Systems,* 14 (6) (2004) 393-405.
51. T. Goldring, Scatter (and other) plots for visualizing user profiling data and network traffic, in *2004 ACM Workshop on Visualization and Data Mining for Computer Security*,eds. (ACM Press, 2004) pp. 119-123.
52. A. Lazarevic, V. Kumar, and J. Srivastava, Intrusion detection: A survey, in *Managing cyber threats: Issues, approaches, and challenges* Springer US, 2005), pp. 19-78.
53. J. H. Friedman and J. W. Tukey, A projection pursuit algorithm for exploratory data-analysis, *IEEE Transactions on Computers,* 23 (9) (1974) 881-890.
54. K. Pearson, On lines and planes of closest fit to systems of points in space, *Philosophical Magazine,* 2 (6) (1901) 559-572.
55. H. Hotelling, Analysis of a complex of statistical variables into principal components, *Journal of Education Psychology,* 24 (1933) 417-444.
56. P. Diaconis and D. Freedman, Asymptotics of graphical projection pursuit, *The Annals of Statistics,* 12 (3) (1984) 793-815.
57. E. Corchado and C. Fyfe, Connectionist techniques for the identification and suppression of interfering underlying factors, *International Journal of Pattern Recognition and Artificial Intelligence,* 17 (8) (2003) 1447-1466.
58. C. Fyfe and E. Corchado, Maximum likelihood hebbian rules, in *10th European Symposium on Artificial Neural Networks (ESANN 2002)*,eds., 2002) pp. 143-148.
59. E. Corchado, Y. Han, and C. Fyfe, Structuring global responses of local filters using lateral connections, *Journal of Experimental & Theoretical Artificial Intelligence,* 15 (4) (2003) 473-487.
60. H. S. Seung, N. D. Socci, and D. Lee, The rectified gaussian distribution, *Advances in Neural Information Processing Systems,* 10 (1998) 350-356.
61. E. Corchado, P. Burgos, M. D. Rodriguez, and V. Tricio, A hierarchical visualization tool to analyse the thermal evolution of construction materials, in *CDVE 2004*,eds. Y. Luo (Springer, Heidelberg, 2004) pp. 238-245.
62. E. Corchado, M. A. Pellicer, and M. L. Borrajo, A mlhl based method to an agent-based architecture, *International Journal of Computer Mathematics (Accepted - In press),* (2009).
63. Á. Herrero, E. Corchado, L. Sáiz, and A. Abraham, Dipkip: A connectionist knowledge management system to identify knowledge deficits in practical cases, *Computational Intelligence (Accepted - in press),* (2009).
64. P. Demartines and J. Herault, Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets, *IEEE Transactions on Neural Networks,* 8 (1) (1997) 148-154.
65. T. Kohonen, The self-organizing map, *Proceedings of the IEEE,* 78 (9) (1990) 1464-1480.
66. E. López-Rubio, R. M. Luque-Baena, and E. DomÍnguez, Foreground detection in video sequences with probabilistic self-organizing maps, *International Journal of Neural Systems,* 21 (3) (2011) 225-246.
67. R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, The 1999 darpa off-line intrusion detection evaluation, *Computer Networks,* 34 (4) (2000) 579-595.
68. R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D.

Wyschogrod, R. K. Cunningham, and M. A. Zissman, Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation, in *DARPA Information Survivability Conference and Exposition (DISCEX '00)*,eds., 2000) pp. 12-26.

69. R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, Analysis and results of the 1999 darpa off-line intrusion detection evaluation, in *Third International Workshop on the Recent Advances in Intrusion Detection (RAID 2000)*,eds. H. Debar, L. Mé, and S. F. Wu (Springer, Heidelberg, 2000) pp. 162-182.
70. Kdd cup 1999 dataset, http://archive.ics.uci.edu/ml/databases/kddcup99/kddcup99.html. Last access: 24/06/2009
71. M. Elkan, Results of the kdd'99 classifier learning contest, http://www-cse.ucsd.edu/users/elkan/clresults.html, 1999.
72. J. McHugh, Testing intrusion detection systems: A critique of the 1998 and 1999 darpa off-line intrusion detection system evaluation as performed by lincoln laboratory, *ACM Transactions on Information and System Security,* 3 (4) (2000) 262-294.
73. S. Maheshkumar and S. Gursel, Why machine learning algorithms fail in misuse detection on kdd intrusion detection data set, *Intelligent Data Analysis,* 8 (4) (2004) 403-415.
74. Euskalert, basque honeypot network, http://www.euskalert.net. Last access: 10/05/2010