Towards an Intelligent Intrusion Detection System based on SOM Architectures

MIGUEL ANGEL PEREZ DEL PINO[1]
Faculty of Informatics and Media
University of Applied Sciences Brandenburg
14770, Brandenburg
Germany

Supervisor: *Dipl.-Inform. Ingo Boersch*

ABSTRACT

With the growth of computer networks in the recent years, security has become a crucial aspect in modern computer systems. An adequate way to detect systems misuse is based on monitoring users' activities and network traffic. Several intrusion detection methods have been proposed but they result in an expensive implementation and doubtful yield. In this paper, it is analyzed an approach based on the application of non-supervised auto-organizing artificial neural networks to improve the performance of real-time intrusion detection systems.

1. INTRODUCTION

Intrusion detection systems and information defence operations have been designed with the intention of protecting availability, confidentiality and integrity of critical information systems connected to intercommunication networks.

From an ideal point of view, an intrusion detection system (IDS) has capacity to detect not authorized real-time accesses, attacks or illegal use of a computer system, minimizing damage and/or blocking the activity by executing specific operations; also allowing to study efficiently the causes and ways used in a certain attack. The two main IDS categories are: 1) the one which includes analyzing network traffic systems; 2) the one which groups systems that analyze audits generated by the operating system.

Several models have been proposed [25]: expert, probabilistic, and heuristic systems, petri networks, and different kind of automata. The implementation and maintenance of this prototypes is highly expensive; just check the quantity of information generated by the monitoring that is needed to be stored to be analyzed afterwards [21, 22].

The needs to reduce the consumption of resources in the analysis process and the number of false positives raise the greater challenges in this research area. To obtain them, it is analyzed a scheme based on auto-organized maps that allows focusing user attacks and, at the same time, analyzing network traffic and other events that could incur the anomalous use of a computer box.

[1] E-mail: *perez-m@fh-brandenburg.de*

2. RELATED WORK

Computer systems security and data confidentiality have become high priorities in the information society needs. Jim Anderson was the first one to enter himself into the concept of computer intrusion in the beginning of the 80's, defining it as "*the potential possibility of a deliberate unauthorized attempt to access and/or manipulate the information, as well as render a system unreliable or unusable*", [5].

2.1. MISUSE DETECTION VS. ANOMALY DETECTION

It is important to differentiate between misusing and anomalous behaviours. The main advantage of the proposed designs to alert misusing is the attempts of known attacks are fast and effectively detected with a very low rate of false positives. It is very simple to detect an attack that has been codified at first on the IDS knowledge base. If log files do not contain relevant signatures of the attempt of attack, no alert action will be taken.

However, the disadvantage arises when we notice ourselves that the model is not able to alert new intrusion techniques. Thus, the rate of false positives* can rise in an extreme form, always depending on the attacker's skill. As consequence, the misuse detection provides little defence to the systems, unless they are able to generalize from well-known attacks.

Anomaly detection methods are indeed enabled to focus their attention in possible new attacks. This is possible due to there is no specific patterns search, but the activity is compared with another happened before. Any activity that is turned aside of a tendency of normal behaviour will be considered like anomalous, and with it, considered as a possible attack. Moreover, anomaly detection schemes are based in users' and system's history to create internal representations; not in predefined patterns. An important detail to consider in this type of techniques is they are not able to identify what kind of attack is being carried out.

The main disadvantage of these methods is their high false alarms rates. Any typical deviation on the behaviour can be labelled as an attack attempt, and commonly, many habits can be on the range defining these possible attacks.

2.2. ARTIFICIAL NEURAL NETWORKS APPLICATION

The employment of neural computing techniques in systems as the analyzed in this paper is recent, taking into account that the first incorporations were made in the beginning of the 90's. The majority anomaly detection related developments based on auto-learning make use of artificial neural networks (ANN), i.e. *Hyperview*, [7]. The usual network traffic is introduced to an AAN which learns traffic patterns. The new traffic, including possible attempts of attack, is applied to the network once its training stage is done, with the purpose of detecting intrusions.

R. Lippmann and *R. Cunningham* from the *MIT Lincoln Laboratory* focused their thesis [7] onto the application of artificial neural networks in misuse detection, looking for specific attack terms in the network traffic using MLP.

J. Cannady and *J. Mahaffey* from the *Georgia Technical Research Institute* set their research [6] into applying MLP and MLP/SOM combination to detect intruders. The researchers proposed specialized neural networks for every protocol involved, converting themselves in a system based on high specialization layers.

* *False Positive: False Alarm Firing. No intrusion.*
False Negative: Non-detected attack attempts.

L. Girardin from the *UBILAB Laboratory* [15] employed SOM techniques to do network traffic clustering and detect attempts of attack. *Girardin* tested this model successfully with the following types of attack: *IP spoofing,* password guessing on FTP, network scanning and networking jump.

The majority of the proposed prototypes have offline functioning basics. Researchers do not often present time representation schemes on their works. *Labib* and *Vemuri* describe in [19] a real-time representation method using Kohonen maps.

2.3. OBJETIVES

The intention of this research has been the study of IDS implementation using self-organizing methods instead of common supervised techniques such as *Perceptrons* (*MLP*) and *Backpropagation*. Basing on *Labib and Vemuri*'s proposal [19], the developed analysis has followed the steps below:

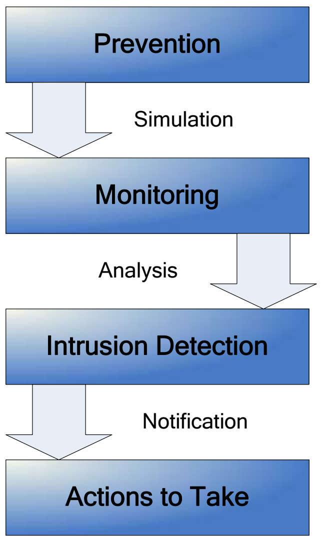
1. Harvesting and generating a training data set which structure follows the details in section 5.3.

2. Applying SOM techniques to detection. Two types of traffic to be detected: *normal* and *anomalous*.

3. Analyzing and testing if the obtained results in a non-controlled environment follow *Labib and Vemuri's* experiment conclusions.

3. INTRUSION DETECTION SYSTEMS ANALYSIS

The main task to be carried out by IDSs is to defend a computer system against attack attempts, repelling them. The detection of hostile attack attempts depends on the number and the adequate type of actions taken. Intrusion prevention requires a good combination of watching and traps, motivated by the investigation of threats that can be suffered, have taken place or potentially will take place.



The distraction of intruder's attention from protected data is an important task to be carried out. The alternative is using trap systems – constantly monitored, together with the compromised system. Data generated by the monitoring task will be carefully checked by the IDS, detecting possible security risks.

Once an intrusion has been detected, the IDS will alert the administration team. The next step will be taken either by the administrators or by the own IDS if it has resources to do it itself – blocking sessions, adding rules to firewalls, backuping, routing connections to trap systems, etc. Every action taken will depend on the enterprise information and systems security policy. Within the different tasks from IDS, the intruder identification is fundamental; it will be always useful in forensic analysis of incidents and will allow the undertaking of immediate actions against him/her, again, depending on the policy of security in the organization.

3.1. INTRUSION DETECTION SYSTEMS ARCHITECTURE

Until recently, the job of IDS was centred on individual and independent systems. However, the most common organization needs to defend a distributed set of computer systems in interconnected LANs. Although it is possible to increase the defence using isolated-on-each host IDS, cooperation and coordination between all the IDS in a network, the experience demonstrates, grants a more effective defence, [28]. *Porras* points important results in the design of distributed IDS:

1. On heterogeneous environments, involved systems will use distinct event loggers, so that a distributed IDS needs to handle with different audit log formats.

2. On distributed architectures, one or more nodes will serve as compiling information points. This implies a data transit over the network between analyzer hosts and satellites. It is highly important to assure reliability and confidentiality of the transmitted information; it is necessary to avoid that an intruder can alter audit data masquerading his/her activities.

3. On centralized architectures, there is only one unique event logger and analyzing point. This makes easier working with audit logs but it creates a potential bottleneck. Moreover, there exists just one fissure point. Decentralizing the architecture, there are several loggers and analyzers, coordinating their jobs and sharing information.

The detailed architecture in Fig.1 shows the distributed IDS scheme developed by the *University of California at Davis* [16, 27, 28]. It consists of three main components:

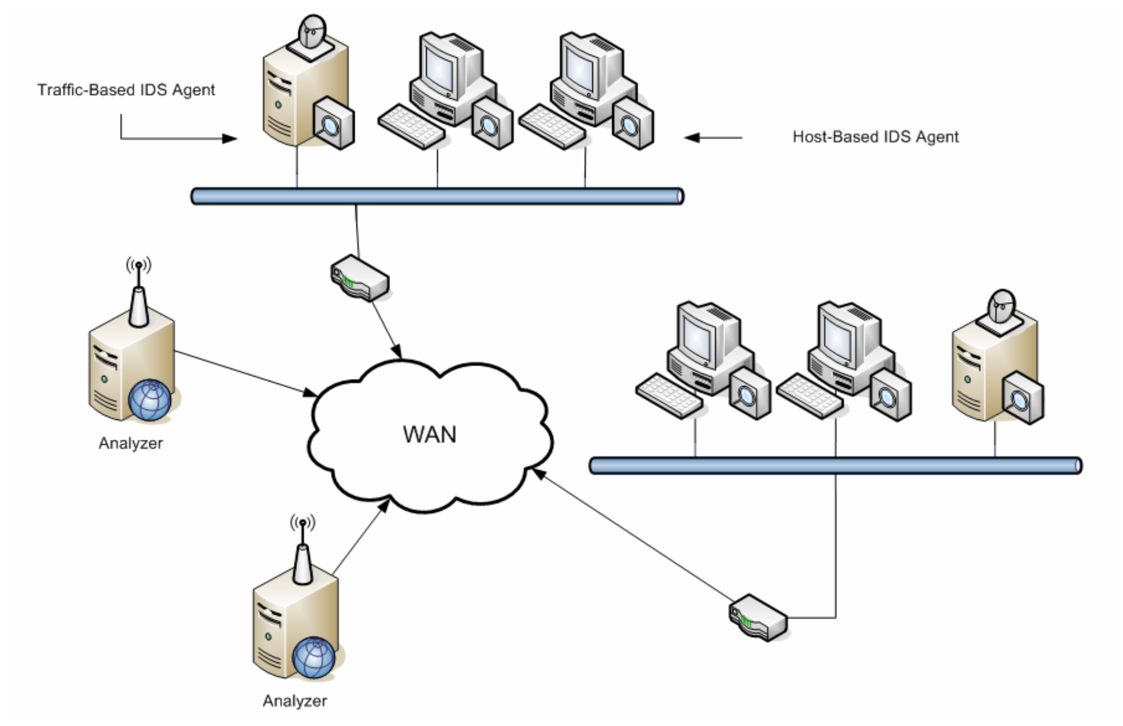


Fig.1. Distributed IDS Architecture

1. *Host-based IDS Agent*: It is an event logger that runs as a subordinated process of the monitored system. It will audit certain security events and will send them to the Analyzer.

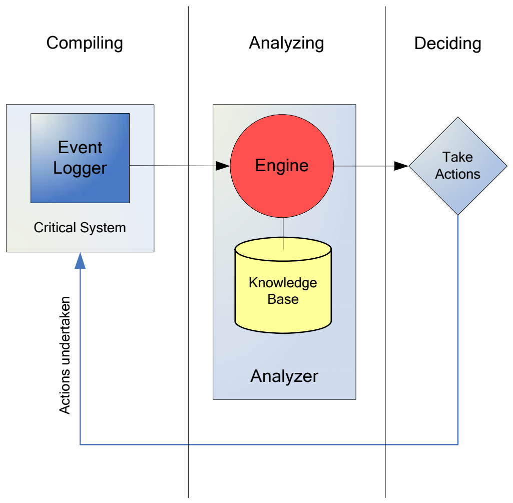
2. *Traffic-based IDS Agent*: It works the same way as the Host-based one but audits local network traffic and sends reports to the Analyzer.

3. *Analyzer Agent*: It receives audit logs and studies them, correlating data and detecting intrusion or attack attempts.

3.2. INTRUSION DETECTION SYSTEMS STRUCTURE

The proposal explained in this paper is a segmented structure where each pipeline stage will be in charge of a partial function of the complete payload. A detailed pipeline for generic IDS is detailed below:

1. *Events Logger*: It consists of specialized software in harvesting user, native processes software and network traffic information. With regard to events based on host, practically all multi-user operating systems have this sort of software integrated. As far as the network traffic, it will be used special software to audit connections host to host, services and the traffic load.



2. *Analyzer*: This stage consists of two basic functional units: the engine and the knowledge base. Its job is identifying when a potential attempt of attack is being carried out. It is usually implemented using expert systems and/or statistical models.

3. *Decisions Module:* It is the stage in charge of alerting the administration team about a possible attempt of attack. It provides relevant information in order to in order to confront and repel the attack or intrusion. It can also take decisions to minimize damage, modifying the behaviour of the involved systems.

3.3. PROPOSED ANALYZER MODELS

Different proposals have been done over the last 25 years, always in search of significant improvements in what relates to attacks and intrusion detection. Since that Anderson's postulate in 1980, expressing the possibility of distinguishing between legitimate user and supplanted user, attacks have happened to be merely local to have a remote foundation; fact motivated mainly by the expansion of the computers communication networks and the Internet.

The statistical anomaly detection model implies user-behaviour-related data compiling in a time period. The statistical tests are applied to observed behaviours to determine with high reliability grade if some user behaviour is or not a legitimate user's, [28].

Usually, the implementation of this technique has two possible slopes:

1. *Profile based*: It is developed a user activity profile for every user and is used to detect changes in the user behaviour on personal accounts.

2. *Threshold based*: There are defined thresholds, independent from the user, which determine the frequency which the different events occur with.

This sort of technique intends to model the user behaviour. Denning has proposed in [9] different approaches that can be adopted to make the corresponding metrics: average and standard deviation, multivariable analysis, time series, Markov chains, and operations based on old registry audits.

It is evident that the users of the system do not behave the same manner in all work sessions, they do not even have to run the system commands in the same order to complete a task. A minimal deviation in the usual behaviour of a subject will throw a security alert; so that, this implies a high false positives rate. On the other hand, an important fissure of security is revealed: a user could run just one command able to execute different harmful processes without being detected by the analyzer, i.e. batch and scripts.

Rules based detection implies a trial to define a scheme to be used in intrusion behaviour determination. It is said that the identification of a penetration in the system is based on an expert system approach, looking for suspicious behaviours to locate to possible attacks and/or intrusions.

With this method, the historical audit events are analyzed to identify the patterns of use and to generate rules that describe those models. The rules can represent behaviour schemes in the past of users, privileges, programs, time slices, terminals, etc.

The present behaviour is then observed and every transaction is related to the set of rules to determine if it adjusts to some observed previous behaviour pattern. According to Stallings in [28], this scheme is based on the observation of last behaviours and, in effect, on assuming that the future behaviours will be like the past one.

So that this approach is effective, an extensive data base is required containing the rules. In the model described by *Vaccaro* and *Liepins* in [29], the data base contains a set of between $10^4$and 10^6 rules.

Garvey and *Lunt* postulated in [12] that the conduct can be defined by means of an expert system of these characteristics and thus, proposed to predict performance patterns. *Lee* and *Stolfo* in [20] exposed a similar development analyzing execution plans using a system of similar features.

Nevertheless, it is thought that even with such an order of rules and a detailed historical study of activities, it is possible to leave undetected illicit behaviours or not to contemplate certain activities.

The revision of [8, 11, 12, 20, 16, 23, 26, 28] is recommended for details on other proposed prototypes.

4. NEURAL COMPUTATION

The human being has always been featured by his constant search of new paths that allow a better problem resolution. Different machines that allow implementing easily algorithms to resolve several problems which resulted hard to resolve before have been born.

However, it is observed an important limitation: certain problems do not admit an algorithmic treatment to be resolved. If it is examined with attention all those problems that cannot be expressed through an algorithm approach, we will observe that all of them have a common feature: experience. The human being is able to resolve these situations using accumulated experience. Thus, it seems to be clear that a way of approaching the problem may consist in the construction of systems that are able to reproduce this human feature.

Neurocomputing is the discipline based on the philosophy of structure and functionality of biological neural networks. On the other hand, neural networks theory, which is based on the volitive and cognitive processes of the human being, tries to mechanize mental procedures. Neurocomputing tries to emulate behaviours such as the learning, the capacity of memorization and information retrieval, and the association of facts with the purpose of solving certain technical problems.

A neural network is a system for data processing, whose basic processing unit is inspired by the fundamental cell of human nervous system: the neuron. All the processes of the human body are related in some or another form to the activity / inactivity of neurons.

Learning can be defined as the capacity of a system to absorb information from its environment without the help of an external agent. Neural networks have the capacity to learn, improving their operation, and to adapt their selves to the specific conditions of a certain scope. This implies that those problems which at first cannot be solved can be solved after obtaining some information about their domain.

Due to their constitution and fundaments, artificial neural networks present a great number of similar characteristics to those of the brain. They are able to generalize from previous cases to new cases, to abstract essential characteristics from inputs that represent irrelevant information, etc. So that they offer numerous advantages; this type of technology is being applied in multiple areas. Among the advantages, it can be listed: adaptive learning, auto-organization, tolerance to failures, operation in real time (parallelism) and easy insertion within the existing technology, designing chips specialized for neural networks that improve their capacity in certain tasks.

5. ANALYSIS OF AN ANOMALOUS TRAFFIC DETECTOR BASED ON ARTIFICIAL NEURAL NETWORKS

5.1. OBJETIVES

The main aim of this paper is to analyze and test the viability of using a model based on a non-supervised learning paradigm by means of artificial neural networks. With the prototypes presented until today’s date, the misuse detection is very accurate as long as the attack being carried out is listed on the IDS knowledge base. Evidently, it is not possible to predict all those illegitimate actions that could arise and, moreover, the attempt to carry out such task is laborious. The anomaly detection is always made necessary in last instance.

Several made studies [24, 20] demonstrate that an IDS performance is increased if, in addition to analysis of cases of attack (misuse knowledge base), a method of *detection of anomalies* is gotten up. This procedure allows, a priori, classifying abnormal actions and/or trafficking like a possible intrusion, locating new forms of attack (for which identification in the knowledge base still does not exist).

5.2. NETWORK TRAFFIC ANALYSIS

For harvesting and sorting the corresponding network traffic data, a sniffer will be used. Sniffers are software tools that capture packets going over a computer network with the aim of monitoring the traffic that circulates around it.

Data harvesting must be made on the basis of some policy. In previous proposals, the sniffer remains a certain amount of time monitoring the network traffic, which will be later analyzed [18]. In order to carry this task out, the information the sniffer has gathered is packed basing on connections. Specifically, a connection is defined as "*a sequence of TCP packets that begins and finishes at certain times, between which the data flows from a source IP address to a destination IP address and over a specific data protocol*" [2].

This model requires harvesting during a defined amount of time for a later offline analysis. In real cases, it is effective-less having to wait several hours, days or weeks to have the information prepared to check if an intrusion or attack attempt has occurred. Many of the researchers do not describe the problem of time representation in their projects. Some of them suppose that the performance of a real-time system can be reached thanks to data processing minimization, which entails simpler designs implementations.

Labib and Vemuri consider in [19] the representation of the temporary variable is an important element when the network traffic is considered: *attacks are carried out by means of a successive series of determined packets destined to a host or hosts in a finite time interval*. Their proposal tries to grant the effect of real time analysis by means of a packets capture procedure based on intervals. They suppose that real time performance can be reached diminishing the processing of data, and such form, using simpler designs.

Even though packets arrival and departure times are explicitly available before the information is pre-processed, the authors of [19] chose not to use explicit time representation because of the reasons discussed in [22]. The implicit time representation scheme used is based on the collection of `*n*' successive packets and the organization of them in a unique input vector that will be given to the neural network. This way, the neural network will be able to analyze `*n*' ordered packets in the same way they arrived the Ethernet port.

5.3. INFORMATION PREPROCESSING METHOD

Labib and Vemuri suppose that a real-time system performance can be achieved using Self Organizing Maps (SOM) thanks to data processing minimization, and thus, implementing simpler designs. They proposed the following structure in order to achieve it: *communication packet unit vectors* compound by well identified elements: *source address*, *destination address* and *protocol involved*. Source and destination addresses will be sectioned into octets, and there will be only employed the last two octets of each address (apparently to reduce the redundancy level in patterns, due to the two upper octets are repeated constantly). The protocol involved will be coded summing the characters decimal values by the ASCII norm[*].

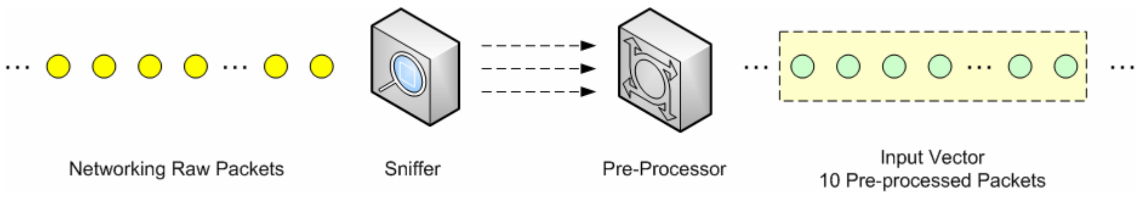


Fig.2. Pre-Processing Procedure.

The structure formed by 10 unit vectors will generate an input vector for the neural network, (Fig.2). This implies that the neural network will receive 50 values in its input layer, and so that, the information contained in 10 networking packets. Labib and Vemuri's intention has been to give an implicit time representation to the developed prototype.

[*] i.e.: *Value (TCP) =ASCII (T)+ASCII (C)+ASCII (P)*

Normalization process has been studied from two sides:

1. Using Euclidean normalization.
2. Looking for the Distribution Gravity Centre (*DGC*) in the [-1, 1] range.

Input vectors will be made up of float values once normalized. It can be observed that using Euclidean normalization generates higher activation values. It is recommended a *DGC*; certainly, the result of this operation notably improves the SOM firing behaviour, being reduced the corresponding signals.

The pre-processing stage has been implemented by a *perl application* which interprets and encodes all the needed tasks described above to convert the sniffer output into an useful dataset to be presented to the neural network; that is to say, to generate the input vectors dataset. This application has been designed to:

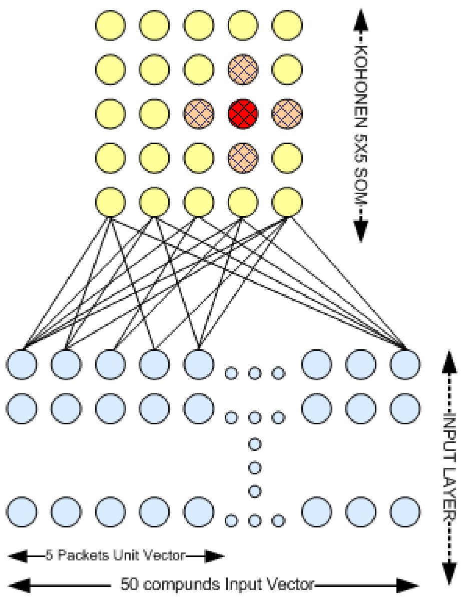
1. Support *offline pre-processing*, so that it can read the file-captured sniffer output and generate an input dataset file.

2. Support *online pre-processing*, receiving and interpreting directly the sniffer output, in real-time.

The main tests to probe the neural network success have been made using pre-processing mode (*1*).

5.4. THE ARTIFICIAL NEURAL NETWORK

Non-supervised learning provides a simple and efficient way to cluster data sets. Real time processing and classifying is highly expensive. Self Organizing Maps (SOM) have been selected due to their fast processing speeds and high conversion rates in comparison to other learning techniques. Even more, SOM preserve the topographic data structure; that is to say: the relationships between *senders*, *receivers* and *protocols* involved.



The model presented in [19] has been reproduced to be tested. It is compound of 50 neurons in the input layer and a 5x5 bi-dimensional Kohonen SOM layer. It has been detected the great importance the selection of a correct weight initialization function has. Different tests have been made using the following functions: a) *Kohonen Random*; b) *Kohonen Pattern Random*; c) *Simple Random*. It is observed that *(b)* produces higher activation values all around the map, while *(a)* and *(c)* are more conservative ones, with lower activation values, also allowing a better pattern clustering. For the final implementation tests made, function *(a)* has been set. The neural network has been simulated using JNNS, [30].

5.5. EXPERIENCE

The employed dataset was harvested using the application '*tcpdump*' under a Sun Solaris operating system. The Computer Science & Systems Faculty network, composed by hundreds of systems making simultaneous transactions was test environment. The sniffer filtered all the connections directed to a critical system. The watched server offers all type of services to clients, such as *remote desktop protocol, SSH, FTP, HTTP, NETBIOS file and print sharing server*. With the purpose of having a heterogeneous dataset, the data harvesting was made at different times: high payload times (*rush hours*) and reduced payload times (*weekends, evenings*). The supposed as '*normal traffic*' had no '*attacking traffic*' on it. The experiment was done in a controlled way. There were detected no harmful or potentially harmful traffic by firewalls in the network used; so that it can be assured the normal traffic was clean. The anomalous traffic is based on *DoS* attacks, made from different network systems and with different denegation features. The gathered dataset including normal and anomalous traffic ascends to 21233 logged transactions.

After the information has been pre-processed, it has been obtained a *training set* of 2122 input patterns. Another little set of harmful connections mixed with normal traffic has been packed in other dataset, to be considered as *validation set*, and being a compound of 100 patterns. Labib and Vemuri do not present the parameters employed in their experiment; this is the reason for different parameter testing. The SOM neural network was trained employing the training dataset with the features below:

1. The used neighbourhood has a 1-size radius parameter. This supposes the adaptation of the network weights will be made between the neurons surrounding the winner one. This is the unique parameter presented in [19].

2. The SOM adaptation parameter, η, set to different values, decreasing from 1 to 0.6.

3. The SOM will be trained during a 1000-cicles period.

4. The decreasing SOM functions (neighbourhood and radius ones) were adjusted to 0.98.

6. EXPERIMENTAL RESULTS

6.1. ANALYSIS OF RESULTS

After the training phase, it is concluded that normal traffic is located in a concrete area on the bi-dimensional map, (Fig.2). The blue area in Fig.2 shows the presence of normal traffic. The neurons involved in such area are the ones with the capability of detecting normal traffic patterns; they are the ones with lower activation values in the map (*so that, the ones with shorter Euclidean distance to the input pattern*) on the presentation of usual traffic patterns.

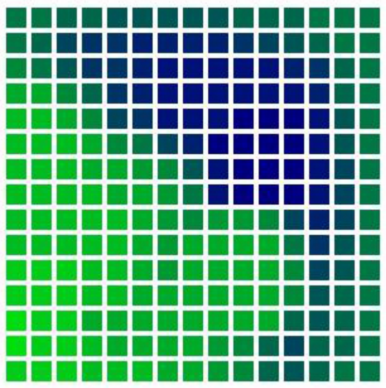


Fig.2. Normal Traffic Response

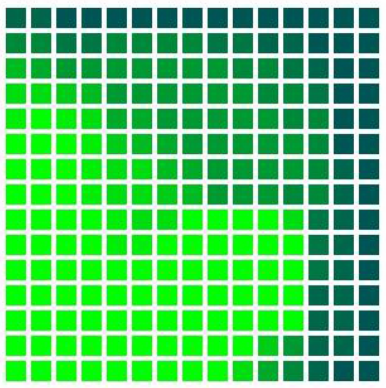


Fig.3. Attacking Traffic Response

However, it is shown in Fig.3 the neural network response when presenting a dangerous traffic pattern. The attacking traffic produces a little higher activation values in the map (*light green area*).

A brief analysis about the functioning mode of the model presented in [19] is commented below:

1. Since the 80% of the content of the input vector is just IP addresses octets, 80% of the training set is made up of practically invariant information. The other 20% of the information used for the network training supposes a limited series of communication protocols employed in the transactions, encoded numerically. Most of the transactions that happen in an Internet family network are catalogued over TCP.

   This means there exists a high redundancy in the patterns showed to the network, which apparently is the main generator of knowledge, but the network is unable to distinguish a normal sequence from an attacking sequence in all the cases (i.e. *An intrusion from the inside network can be considered as a non attempt of attack*).

   It can be said that the network learns to identify traffic coming in and going out from the computer network whose packets were used in the training stage. Taking into consideration the previous ones, activities can be confused.

   The author of this paper thinks the neuronal network does not need to know the source and destination IP addresses (nor a section of them) to detect a possible attack or intrusion in the system. The presence of just segmented addresses lacks useful information to focus on in case of some kinds of intrusion attempts.

2. It should be taken into account that the attacking traffic has been made up of extensive amounts of ICMP packets. Due to signals overlapping of normal and attacking traffic in the validation set, entrance vectors are not strange enough to the network - according to the proposed encoding and structure for the input vector in [19].

   If it is only thought about having ICMP connections in an input vector (*50 components; 10 ICMP packets in sequential order)*, the network could detect a certain difference with the patterns of normal traffic that it knows, but evidently, this it is not the real case. The attacked system is serving at the same time that it is attacked.

3. It does exist neither a sequences control nor some method with historical intention; in other words, one does not have any procedure to analyze what has happened, is happening and/or to predict what is going to happen in the early future. Thus, other sort of attacks such as sequence numbers one and its evolution, the *SYN flood*, will not be detected.

6.2. TECHNIQUES INTO CONSIDERATION

It has been thought about possible tendencies in the research and test line. As an effect of the comments presented in section 6.1, the data encoding proposed at [19] is not considered to be useful at all by the author of this paper. It is thought that the information packed in the input vector cannot detect possible attempts of attacks made from the inside network. It also lacks of useful information to detect complex attacks.

Two different approaches are proposed:

- The first approach is based on a mixed *training data set of normal and attacking traffic*. This means the knowledge of what kind of attacks the network should be able to recognize, and though, the network is being limited to respond against unknown types of attack.

  Being faithful with the comments presented in the introductory section 1, the related work section 2, and the overview analysis in 5.1, this is not the line wanted to be followed.

- The second proposal is based on *using a clean and extensive normal traffic data set to train the neural network*. Normal networking traffic will occupy the whole SOM concerning their distribution; so that on an attempt of attack, the normal-traffic-specialized neural units will fire with higher activation values. It must be studied the quantization vector between the input pattern and the winning neuron to announce the appearance of an attempt of attack.

  The author of this paper thinks this has been the Labib and Vemuri's goal on the developed system proposed in [19]. By the way, this line seems to be more faithful to the comments presented in sections 1, 2.1 and 5.1

7. CONCLUSIONS & FUTURE WORK

It has been tried to reproduce Labib and Vemuri's NSOM model, concluding that the information encoded to make the network categorize different types of traffic is not enough. Logically, the simplicity of the model presented in [19] will have to be reduced because it is not effective in real situations. It is considered as needed to represent more and more heterogenic information to allow the neural network making a more accurate traffic clustering.

It can be seen that normal traffic has a set of defined connections, with a limited data load, over defined protocols and carrying specific services tasks, always limited in time. That is what makes one think that using just IP addresses and/or the protocol involved in the transaction is not enough; there is more useful information in the header of packets being trashed that can improve performance and clustering.

Moreover, two approaches have been considered in 6.2. The author of this paper's opinion is near to employ the second proposal; that is, to calculate the quantization error to detect high distances on a normal-traffic-trained SOM.

In future work, it is planned to accomplish two main items:

1. To study and propose a good encoding data vector, which can make the neural network to extract special attack features from traffic data. It is thought that this will be found in the flags compounding the packets.

2. To search and implement a more appropriate SOM structure, studying the different parameters involved, depending on the detection problem being performed. The size of dimension employed seemed to be useful to test and see Labib & Vemuri's model, although it is needed to study and test the other parameters which affect in the SOM performance.

3. To employ another simulation tool rather than JNNS. The author's intention is to develop an own SOM implementation to test the model.

8. REFERENCES

[1] The 1998 intrusion detection off-line evaluation plan.
URL: *http://www.ll.mit.edu/IST/ideval/index.html*.

[2] Knowledge discovery in databases DARPA archive. Task Description. 1999.
URL: http://www.kdd.ics.uci.edu/databases/kddcup99/task.html.

[3] *Microsoft Security Bulletin Summaries*, in M. Corporation. 2002, 2003, 2004, 2005.

[4] R. Hecht-Nielsen, *Neurocomputing*. Addison-Wesley Publishing. 1989.

[5] J. Anderson, *Computer Security Threat Monitoring and Surveillance*, James P. Anderson Co. 1980. Fort Washington, PA, USA.

[6] J. Cannady, Mahaffey, J., *The application of Artificial Neural Networks to Misuse detection: initial results*, in I. P. o. t. N. I. S. S. C. (NISSC'98), ed., Georgia Tech Research Institute, 1998.

[7] R. Cunningham, Lippmann, R., *Improving Intrusion Detection performance using Keyword selection and Neural Networks*, *MIT Lincoln Laboratory*, MIT Lincoln University, 1999.

[8] H. Debar, Becker, M., Siboni, D., *A neural network component for an intrusion detection system.* In Proceedings of 1992 IEEE Computer Society Symposium on Research in Computer Science Security and Privacy, pp. 240-250, 1992.

[9] D. Denning, *An Intrusion-Detection Model*, IEEE Transactions on Software Engineering. 1987.

[10] S. Forrest, Warrender, C, Pearlmutter, B., *Detecting Intrusions Using System Calls: Alternative Data Sets*, In Proceedings of the IEEE Symposium on Security and Privacy, pp. 133-145. 1999.

[11] K. L. Fox, Henning, R. R., Reed J. H., Simonian, R., *A neural network approach towards intrusion detection*. In Proceedings of 13th National Computer Science Security Conference, pp. 125-134. 1990.

[12] T. D. Garvey, Lunt, T. F., *Model-based Intrusion Detection: Current and future directions.* In Proceedings of 14th National Computer Science Security Conference. 1991.

[13] a. K. Ghosh, Schwartzbard, A., Schatz M., *Learning program behavior profiles for intrusion detection*, In Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring. 1999.

[14] a. K. Ghosh, Schwartzbard, A., Schatz M., *Using program behavior profiles for intrusion detection*, In Proceedings of the SANS Intrusion Detection Workshop. 1999.

[15] L. Girardin, *An eye on network intruder-administrator shoot-outs*, In Proceedings of the 1st Workshop on Intrusion Detection and Network Monitoring (ID'99). Santa Clara, CA, USA. 1999.

[16] L. Heberlein, Mukherjee, B., Levitt, K., *Internetwork Secutiry Monitor: An Intrusion-Detection System for Large-Scale Networks*, In Proceedings of 15th National Computer Science Security Conference. 1992.

[17] The UCI KDD Archive, 1999. URL: http://kdd.ics.uci.edu.

[18] H. G. Kayacik, Zincir-Heywood, A. N., Heywood, M. I., *On the Capability of a SOM based Intrusion Detection System*.Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada.

[19] K. Labib, Vemuri, R., *NSOM: A Real-Time Network-Based Intrusion Detection System Using Self-Organazing Maps*.Department of Applied Science, University of California, Davis, USA.

[20] S. Lee W., S. J., *Data Mining Approaches for Intrusion Detection*. 2001.

[21] P. Lichodzijewski, Nur Zincir-Heywood, A., Heywood, M. I., *Dynamic Intrusion Detection Using Self-Organizing Maps*, 14th Annual Canadian Information Technology Security Symposium Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada. 2002.

[22] P. Lichodzijewski, Nur Zincir-Heywood, A., Heywood, M. I., *Host-Based Intrusion Detection Using Self-Organizing Maps* .Faculty of Computer Science, Dalhousie University, Halifax, NS, Canada. 2002.

[23] R. Miikkulainen, Ryan, J., Lin, M. J., *Intrusion Detection with Neural Networks*, Neural Information Processing Systems, 10.Cambridge, MA, MIT Press. 1998.

[24] R. K. Nichols, *ICSA Guide to Cryptography,* pp. 71-75. 1999.

[25] J. P. Planquart, *Application of Neural Networks to Intrusion Detection*, GSEC Certification v. 1.2d. SANS Institute. 2001.

[26] P. A. Porras, Ilgun, K., Kemmerer, R.A., *State Transition Analysis: A rule-based intrusion detection approach*, IEEE Transactions on Software Engineering, SE, pp. 181-199. 1995.

[27] S. Snapp, *A System for Distributed Intrusion Detection*, In Proceedings of COMPCON. 1991.

[28] W. Stallings, *Fundamentos de Seguridad en Redes, Aplicaciones y Estándares*, 2003.

[29] H. Vaccaro, Liepins, G., *Detection of Anomalous Computer Session Activity*. In Proceedings of the IEEE Symposium on Research in Security and Privacy. 1989.

[30] Dept. of Computer Science of the University of Tübingen, *JNNS: Java Neural Network Simulator*.
URL: *http//www-ra.informatik.uni-tuebingen.de/software/JavaNNS/*