Artificial Immune System Based Intrusion Detection: Innate Immunity using an Unsupervised Learning Approach

[1]*Farhoud Hosseinpour,* [2]*Payam Vahdani Amoli,* [3]*Fahimeh Farahnakian,* [4]*Juha Plosila and* [5] *Timo Hämäläinen*
[1 Corresponding Author, 3 and 4] *Department of Information Technology, University of Turku, Finland.*
{farhos;fahfar;juplos}@utu.fi
[2 and 5]*Faculty of Information Technology, University of Jyväskylä, 40100, Jyväskylä, Finland.*
[2] *pavahdan@student.jyu.fi*
[5] *timo.t.hamalainen@jyu.fi*

Abstract

This paper presents an intrusion detection system architecture based on the artificial immune system concept. In this architecture, an innate immune mechanism through unsupervised machine learning methods is proposed to primarily categorize network traffic to "self" and "non-self" as normal and suspicious profiles respectively. Unsupervised machine learning techniques formulate the invisible structure of unlabeled data without any prior knowledge. The novelty of this work is utilization of these methods in order to provide online and real-time training for the adaptive immune system within the artificial immune system. Different methods for unsupervised machine learning are investigated and DBSCAN (density-based spatial clustering of applications with noise) is selected to be utilized in this architecture. The adaptive immune system in our proposed architecture also takes advantage of the distributed structure, which has shown better self-improvement rate compare to centralized mode and provides primary and secondary immune response for unknown anomalies and zero-day attacks. The experimental results of proposed architecture is presented and discussed.

Keywords: *Distributed intrusion detection system, Artificial immune system, Innate immune system, Unsupervised learning*

1. Introduction

Anomaly-based intrusion detection systems (IDS) have been broadly researched as defensive techniques to address the detection of unknown or zero-day attacks. Unlike misuse-based or signature-based types of IDS, which take advantage of the predetermined signature of known attacks, anomaly-based IDS deals with the detection of new types of attack that are unknown to the system. This process is done by detecting variation in the systems' behavior from a previously defined normal system profile. However, it is subject to false alarms as a result of the difficulty in defining the normal state during training. An increasing detection rate with fewer false alarms became an important challenge in the design of anomaly-based IDS.

The artificial immune system (AIS) comprises promising techniques in the form of biologically inspired computing that is applied to solving various problems in the information security field. The AIS is inspired by the human immune system (HIS), which has the ability to distinguish internal cells and molecules of the body from foreign pathogens, so called self and non-self respectively, and protects the body against diseases [1]. In the human body the HIS mainly does this without any prior knowledge of attacking pathogens and their structure. As self and non-self discrimination is a significant attribute in the AIS, it is proposed that it is utilized in designing efficient anomaly-based IDS [2]–[4]. The AIS suggests a multi-layered protection structure for protecting computer networks against attack, like HIS protection against foreign pathogens in the human body [5]. This protection is accomplished through *Innate* or *Adaptive* mechanisms. Innate immunity is immediate; it is the first line of defense for the HIS and provides non-specific protection. Therefore, it has no prior knowledge of specific outsiders. The adaptive immune response, on the other hand, is antigen-specific and is trained using a pre-defined profile of specific attacks [6]. Adaptive immunity also includes a "*memory*" that makes future responses against a specific antigen more efficient [7].

Like other anomaly-based detection techniques, the AIS also takes advantage of monitoring variations of the system's behavior according to a pre-defined normal activity profile as an adaptive immune mechanism. This is done through a learning phase in which a data set containing these profiles is utilized for this purpose. Hence, the efficiency of anomaly detection in the AIS is highly dependent on the learning data set. Substantial research has been conducted so far in the improvement and utilization of AIS-based IDS, the majority of which have utilized a pre-defined and offline data set as learning data for training the IDS. This will reduce the efficiency of the IDS by limiting its knowledge-base to that particular learning data set. Moreover, it is extremely difficult to create a data set of self samples with all variations. In order to cope with this problem, in this paper we have proposed an innate immune mechanism by using unsupervised learning methods as the first line of defense in AIS-based IDS. The innate immune system in our proposed architecture provides online and dynamic categorization of network flows to self and non-self, which is then used by the adaptive immune system to generate attack-specific detectors.

Machine learning methods can be organized based on the type of input available during training. There are three main categories of machine learning: supervised, semi-supervised and unsupervised algorithms. Supervised machine learning algorithms need to be trained by labeled data to distinguish the normal and abnormal behavior of the network. Semi-supervised machine learning algorithms can be trained by attack-free unlabeled data. The acquisition of labeled data from security experts, or finding attack-free data sets for both supervised and semi-supervised techniques, is costly. Recent studies showed the feasibility of unsupervised learning approaches in IDS in comparison with supervised or semi-supervised learning-based IDS. Unsupervised machine learning techniques formulate the invisible structure of an unlabeled data set without any prior knowledge. Clustering algorithms put objects based on their similarities into a cluster or clusters. Clustering algorithms have been used for unsupervised IDS to classify the behavior of the network [8]–[10].

The remainder of the paper is organized as follows. In section **2** we briefly review the AIS and unsupervised learning approaches. We discuss some of the most important related works in section **3**. Section **4** presents the proposed AIS-based IDS. Two main engines of the proposed DIS are explained in Sections **5** and **6**. In section 7, the experimental results of proposed architecture are presented and, finally, we discuss and conclude in Section **8**.

2. Background

In this section, we explain briefly the AIS and clustering method employed in our proposed IDS.

2.1. Artificial Immune System

The human immune system defends the human body against harmful and previously unseen foreign cells using lymphocyte cells. The foreign cells are called antigens, such as bacteria and viruses [1]. The artificial immune system is designed for the computational system and inspired by the HIS; it is applied to solving various problems in the field of information security, particularly intrusion detection systems [11], [12]. Moreover, it incorporates many attributes of the HIS, including diversity, error tolerance, dynamic learning, adaption and self-monitoring [3]. The AIS has the capability to differentiate between the “self” (cells that are owned by the system) and “non-self” (foreign entities to the system) as intrusions. Likewise, detectors similar to lymphocytes are deployed in computer system nodes to intercept and report any malicious activities.

The HIS employs a negative selection process to generate mature immune system cells called T-cells. Forrest et al. [2] proposed a negative selection algorithm to utilize this process of the HIS for a sophisticated anomaly-detection process. This process allows the detection of previously unseen harmful cells without any definition of specific harmful cells.

The algorithm includes three phases: defining self, generating detectors and monitoring the occurrence of anomalies. In the first phase, it establishes the normal behavior patterns of a monitored

system to define "self". It regards the profiled normal patterns as "self" patterns. In the second phase, it generates a number of immature T-cells with random patterns that are compared to each self pattern defined in the first phase. If any generated pattern matches a self pattern, the pattern fails to become a detector and is thus removed. Otherwise, it becomes a mature T-cell detector and is utilized for monitoring subsequent profiled patterns of the monitored system. During the third phase, if a T-cell detector matches any newly profiled pattern, it is then considered that that new anomaly must have occurred in the monitored system.

2.2. Clustering Methods (Unsupervised Machine Learning)

There are several approaches to unsupervised machine learning. Clustering is one of the unsupervised machine learning techniques that have been used for IDS. Clustering techniques group (cluster) samples of data sets based on their similarities to find the outliers as the anomaly. Cluster association and centroid distance techniques are the two most important categories of clustering for anomaly detection. Two popular clustering approaches that have been applied in many proposed IDS are as follows:

1) Density-Based Spatial Clustering of Applications with Noise (DBSCAN) finds a number of clusters starting from the estimated density distribution of the corresponding samples. It requires two parameters: maximum radius of the neighborhood *(ε)* and minimum number of samples required to form a cluster (*minPts*). DBSCAN detects a density-connected cluster by discovering one of its core samples, *p*, and computes all samples that are density-reachable from *p*. It checks the *ε*-neighborhood of each sample *p*, *Nε(p)*, in the data set. If the *Nε(p)* of sample *p* consists of the least *minPts* samples, a new cluster containing all samples of *Nε(p)* is created.

2) K-means partitions the given data set into n clusters, in which each cluster has a cluster center (centroid). Any sample assigned to each cluster has a minimum distance to the centroid of the cluster. The Euclidean distance can be used to determine the distance between each sample and the centroid.

3. Related Works

Farmer et al. [13] put forward a new link between biological and computing sciences by proposing the artificial immunology model. Forrest et al. [14] proposed the most effective idea in the utilization of immunity in computer security for self and non-self discrimination. Following this work, they presented basic architecture [3] for the artificial immune system and took advantage of it in deploying the first AIS-based IDS, which was called LISYS. So far different frameworks have been presented in the utilization of the AIS for IDS. However, there are essentially two main approaches to applying AIS in DIS. One approach is classical self/non-self discrimination and another is the application of danger theory as a substitute for the former.

- In [14], a negative selection algorithm is proposed to discriminate between self and non-self entities. The algorithm first creates a set of detectors randomly and then compares it with a set of normal sets (self). If any detectors are matched with any self entity, the system eliminates them and the rest will be kept.
- As a substitute to self/non-self discrimination, the Danger Model was proposed by Matzinger [15], [16]. According to this hypothesis the main cause of an immune response is that a pathogen harms the system and thus is dangerous and not unknown to the system. Aickelin et al. [17] stated that in the IDS paradigm the danger is sensed and measured automatically after a number of intrusions because of the damage caused by the attack. Once a danger signal is detected, it will be transmitted to the nearest artificial antibody around the danger area.

In [5] a multi-layered structure consisting of detection, defense and user layers was proposed. Dal et al. [18] proposed a model in which the primary immune response is evolved through

genetic algorithm to a secondary immune response with optimized detectors that are correspondent to memory cells in natural immune systems. This work was proposed as an enhancement to Forrest and Hofmeyr's work. However, their model still had the disadvantage of central processing with high processing overheads for large network traffic. In our previous work [19], we enhanced their model by proposing a distributed framework to reduce the processing overheads and to increase the efficiency of the IDS. Moreover, the distributed nature of this model resulted in a greater self-improvement feature for this IDS. This work, however, utilizes offline learning data for training the IDS. Since the network behavior is changed in a dynamic fashion, a new profile of normal and abnormal activity needs to be trained to the system dynamically. In order to solve this problem, in this paper we have proposed an innate immune mechanism by utilization of unsupervised learning methods to the primary detection of self and non-self flows, which features online and dynamic training of the adaptive immune system of AIS-based IDS.

Researchers have applied unsupervised machine learning algorithms in IDS to overcome the issues of training and detecting new attacks. For instance, in [20] they proposed a practical real-time solution for NIDS to detect known and unknown network attacks using unsupervised neural networks. They applied several neural networks to improve the detection rate of intrusions. In [8] they also proposed an unsupervised NIDS, which uses different clustering algorithms to detect attacks such as DOS/DDOS, Worm and Network scanning. In [9] they presented a tree-based subspace clustering technique for unsupervised NIDS in high-dimensional data sets. In the proposed model they have generated and analyzed cluster stability for each cluster by using an ensemble of multiple cluster indices. They have also introduced a multi-objective cluster labeling technique to label each stable cluster as normal or anomalous.

4. Proposed Intrusion Detection System

Figure 1 shows the proposed intrusion detection system, which consists of two main engines. The clustering engine performs network traffic clustering into the self or non-self clusters through unsupervised learning techniques. The AIS engine consists of agents that cooperate for intrusion detection. The term “agent” originally comes from Artificial Intelligence (AI) and refers to anything that can view its environment through sensors and act upon that environment using actuators . In this paper, the term agent refers to software agents. Compared to Dal et al.'s [18] work, in our previous work we proposed a distributed model in which we experimented increased performance and efficiency of these IDs as a result of a greater self-improvement rate compare to a centralized structure. This is due to generation of new memory cells and their dynamic synchronization and distribution to all hosts, and thus an enhanced secondary immune response.

The AIS engine trains the primary detectors generated by the negative selection algorithm based on received information from the clustering engine. Moreover, it improves the performance of primary detectors according to the intrusion report analysis from all hosts. In the architecture, the packet pre-processing module is responsible for extracting several attributes from the network traffic to create network flows. These attributes are selected based on the protocol types shown in Table 1.

Table 1. Network Flow specification for each type of packet

Packets Protocol	***Features***
IP	Source IP Address, Destination IP Address, Time of the First Packet, Time of the Last Packet, Duration
TCP	Source Port Number, Destination Port Number, Number of Packets, Number of SYN Packet, Number of SYN-ACK packet, Number of RST Packet, Number of RST-ACK Packet, Number of FIN-ACK Packet
UDP	Source Port Number, Destination Port Number, Number of Packets
ICMP	#Eco Request, #Eco Reply

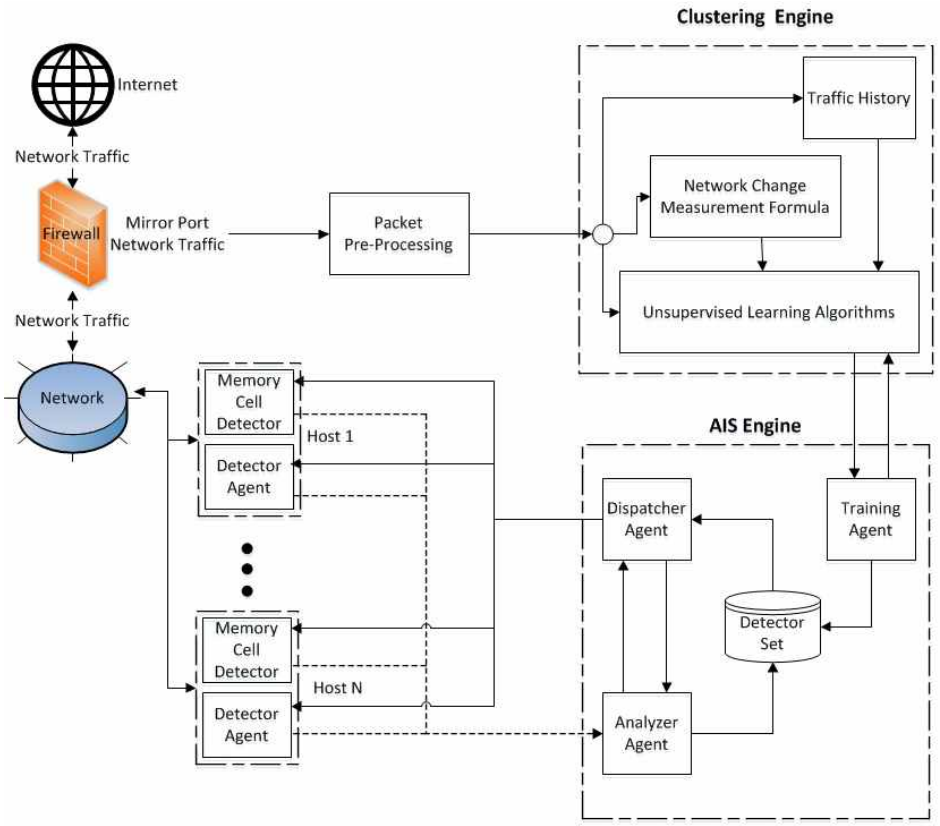

Figure 1. Proposed system architecture

5. Clustering Engine

In order to detect unseen intrusions without using any prior knowledge (training by labeled traffic or signature), we propose a clustering engine as innate immune response. The clustering engine utilizes the DBSCAN clustering method to group the real network traffic into clusters and consider them as self, while behaviors outside of the clusters will be considered as noise or non-self. For this purpose, the engine continuously compares the number of network flows, in different network resolutions (subnets of /0, /8, /16, /24), with a threshold which is dynamically calculated by our proposed network measurement formula in Table 2. Since high speed networks have larger amount of traffic, there is a significant possibility of losing the sign of network attacks. To overcome this issue the system will also monitor the behavior of the network in small resolutions to decrease the possibility of fading the attacks in the normal traffic.

To obtain an accurate threshold, the system needs to determine the previous behavior of the network. It is possible that small attacks to be fade with occurrence of heavy attacks, thus we have applied standardization on the number of network flows by using logarithm (Log) to increase the probability of detecting small attacks during the occurrence of heavy attacks. To determine changes in the network traffic, the system will calculate the "standard deviation" of the number of network flows in different windows from last minute of the traffic. As shown in Table 2 the previous 60 seconds of traffic is divided into four 15 seconds windows. For instance, δ_1 is the standard deviation of number of network flows in the first window which is from the last 65 seconds to the last 50 seconds of the previous network traffic. As it has been seen in so many datasets, it takes 2 to 3 seconds from starting time of the network attacks (such as DOS/DDOS attacks) till its own peak. To minimize the effect of early traffic of the attacks on the threshold, the network measurement formula considers a 5 seconds gap

between every one minute of traffic to calculate the threshold for the current traffic. Sum of the highest standard deviations (from δ_1 to δ_4) and the heaviest traffic from the last minute of traffic can determine the highest traffic which could be accepted as normal. The following equation represents the network measurement formula which calculates the network traffic threshold "T_{nt}":

$$T_{nt} = \left(M\,ax\{\delta_i | i = 1 \cdots 4\} + M\,ax\left\{X_j \log X_j \,\middle|\, j = 1 \cdots 60\right\}\right) \times \gamma$$

Where δ_i is standard deviation of number of network flows in i_{th} window and X_j is number of network fellow in j_{th} second of last minute's traffic. And γ is a coefficient value which can be set to determine the final threshold.

Table 2. Elements of network measurementformula

Last Minute Traffic				***Gap***	***Current Traffic***
Window 1	Window 2	Window 3	Window 4		T_{nt}
δ_1	δ_2	δ_3	δ_4		
65-50	50-35	35-20	20-5	5-0	
$X_j \log X_j$					

The cluster engine considers a network behavior as abnormal when the number of network flows is more than the calculated threshold. Whenever the number of network flows passes the threshold, the clustering engine will cluster the number of in-bounded and out-bounded network flows for each IP (from the past five minutes) to define the normal cluster and obtain an accurate "ε" as the maximum radius of the neighborhood in DBSCAN. Afterwards it will cluster the suspicious traffic with "ε" to find and flag the outliers (noise) as intrusion or non-self. Network intrusions such as DOS, DDOS, Spamming, Worm and UDP storm, generate a large number of network flows. We consider the number of inbound and outbound network flows as the other two essential features for clustering in order to detect these types of intrusion as outliers. Using the aforementioned features of network flow for the clustering engine increases the detection rate of network attacks. Moreover, it will reduce the time and space complexity of clustering algorithms compared to the previous proposed IDSs, which deals with data extracted from packets and bytes (payload). Detected intrusions will be marked as a non-self and set of marked flows will be forwarded to the training agent in the AIS engine to train the primary detectors.

6. AIS-based IDS

The proposed AIS-based IDS has two components: the AIS engine and the host side detectors. This section describes these components in the AIS-based IDS.

6.1. AIS Engine

In the AIS engine, we proposed three agents to employ the adaptive immune response of the AIS in IDS.

Training agent: first converts the network flow information into the binary string with a total 112-bit length as a flow profile (Table 3). Then, using the negative selection algorithm it generates and trains the primary detectors. The negative selection algorithm first generates a number of random detectors (immature detectors) and then trains them with samples of labeled flows from the cluster engine. If each immature detector matches each self sample of the data set, then the system will discard it and generate another in its place. After checking all immature detectors with all self samples, the remaining detector sets undergo the next step of the negative selection algorithm and become mature detectors. Each mature detector will be checked with all non-self samples of labeled flows. If a mature detector fails to match with

some non-self samples, the system will discard this detector; otherwise, this detector will be added to a detector set. This process will continue until all non-self packets are matched with at least three mature detectors.

The negative section algorithm uses the r-Contiguous matching bit role [21] to check the matching between two strings. In this method, two strings are matched if they have at least *r* contiguous identical bits. Finally, the output of the negative selection algorithm is a set of primary detectors, which are archived and synchronized in detector set repository and then sent to the dispatcher agent for distribution to local hosts. These detectors are analogous to primary immune response in the HIS.

Table 3. Depiction of fields in flows profile strings

Name of the Field	Minimum and Maximum Value	Binary Strings Length (bits)
Destination IP Address	0.0.0.0 - 255.255.255.255	32
Source IP Address	0.0.0.0 - 255.255.255.255	32
Destination Port No	0 – 65535	16
Duration	0 – 65535	12
Protocol	0 – 65535	4
Source Port No	0 – 65535	16

Dispatcher agent: distributes detectors from the detector set to all hosts in the network. Moreover, it has the responsibility to communicate with all hosts and synchronize them according to changes in the detector set. It also forwards the reported flow from the memory cell detectors and detector agents to the analyzer agent.

Analyzer agent: employs the proposed genetic algorithm in [18] to evolve the highly fit detectors activated when an anomaly has been encountered. The suspected flow reported from the host's (discussed later) profile of activated detectors and their affinity with reported flow are analyzed in this agent, and an optimized detector, called the memory cell detector, is generated. A memory cell detector is a high-affinity and attack-specific detector with a higher detection ability and analogous to secondary immune response in the HIS [7].

A selection operation is undertaken on activated detectors to select the detectors with the highest affinity for cloning and formation of primary population for genetic algorithm. Those detectors having a fitness value greater than or equal to cloning threshold undergo cloning. The cloning threshold is set as follows.

$$\text{Cloning Threshold} = \frac{\sum_{i=0}^{n} \text{Fitness of detectors}}{n}$$

Where “n” is the total number of activated detectors.

Winner detectors that consist of cloned detectors and remain activated detectors are subjected to the genetic operators of Mutation, Crossover, and Reproduction, which facilitates the evolution of these detectors. This process is repeated and continued for a few generations until a detector with a fitness value higher than all winner detectors is generated. The optimized detector from the genetic algorithm is treated as a memory cell. Finally, the agent sends the memory cell to the dispatcher agent and adds it to the memory cell detector set.

6.2. Host Side Detectors

In order to improve the performance of IDS, detectors are distributed in all hosts in the network rather than a centralized structure. Moreover, the proposed distributed approach is robust and

extendable. All inbound and outbound network flows are checked using these detectors. In each host, we consider two detectors as follows.

Detector agents: comprise a set of trained detectors that have the ability to discriminate between self and non-self flows. These detectors are non-specific and responsible for primary immune response for anomalies that occur for the first time. If a flow matches a detector with an effective affinity, that detector is considered an activated detector and the flow is suspected as an intrusion. To improve the accuracy of detection and reduce the false-positive errors in IDS, we have defined an intrusion threshold (T_i). If the number of activated detectors by a suspected flow is more than T_i, the flow is detected as an intrusion and its information is forwarded to the analyzer agent through a dispatcher agent.

Memory cell detectors: composed of a set of optimized detectors generated by the analyzer agent. As the secondary response of the AIS, memory cells have more accurate intrusion detection abilities. Hence, any flow that activates any of these detectors is treated as an intrusion and blocked by the hosts.

6.3. Detector Life Cycle and Non-self Updater

In order to maintain the efficiency of detectors, we propose to define a lifespan to eliminate unused or weak detectors. Due to machine learning errors, there is a possibility that some of the generated detectors have insufficient detection ability and remain inactivated during their lifespan. Such detectors have negative overheads to the system and reduce its performance. Therefore, in order to solve this problem we define a lifespan for all detectors, during which the number of times the detector is activated is counted. When the lifespan ends, if the counter is less than a threshold, the detector will be discarded and its profile forwarded to a clustering engine as feedback to improve its accuracy. Otherwise, the lifespan will be reset and the detector will remain in the main detector sets.

7. Experimental Result

To evaluate the efficiency of two popular clustering algorithms, we utilized KDD-Cup 99 data set, which is extracted from DARPA-98 traffic network. The number of samples of data set was 22545, which was sufficient for performing the evaluation and comparison between DBSCAN and K-means. The parameters for the different algorithms used are tabulated in Table 4. These parameter values were obtained in a series of preliminary experiments. Table 5 shows the achieved results of K-means and DBSCAN algorithms. We have measured the False Positive Rate (FPR), True Negative Rate (TNR), Accuracy (ACC) which estimated by dividing the total correctly classified positives and negatives by the total number of samples, Recall (REC) or True Positive Rate which estimated by dividing the correctly detected anomalies and the total number of anomalies, Precision of Anomalies (PREC) or positive predicted value which estimated by dividing the correctly classified positives by the total predicted positive count and finally the F1 score, which is the weighted average of the precision and recall. In this experiment, due to ability of DBSCAN for finding arbitrarily shaped cluster, this algorithm demonstrated a better rate of detection compared to K-mean.

Table 4. The parameters of the evaluated clustering approaches

n	***minPts***	***ε***
15	34.4%	1.09

Table 5. The comparison between clustering approaches

Algorithm	***FPR***	***TNR***	***ACC***	***REC***	***PREC***	***F1***
DBSCAN	0.008	0.991	0.771	0.589	0.987	0.738
K-Mean	0.156	0.843	0.607	0.431	0.788	0.557

In our experiment all of the parameters for the network traffic threshold (T_{nt}) is obtained automatically and we only set "γ" as 5. As shown in figure 3, during network attacks the behavior of network passes the threshold. For instance the network traffic during POD (Ping of Death) attack was 10 times more than the threshold or in DDOS (Distributed Denial of service) attack it was 6 times more than the threshold.

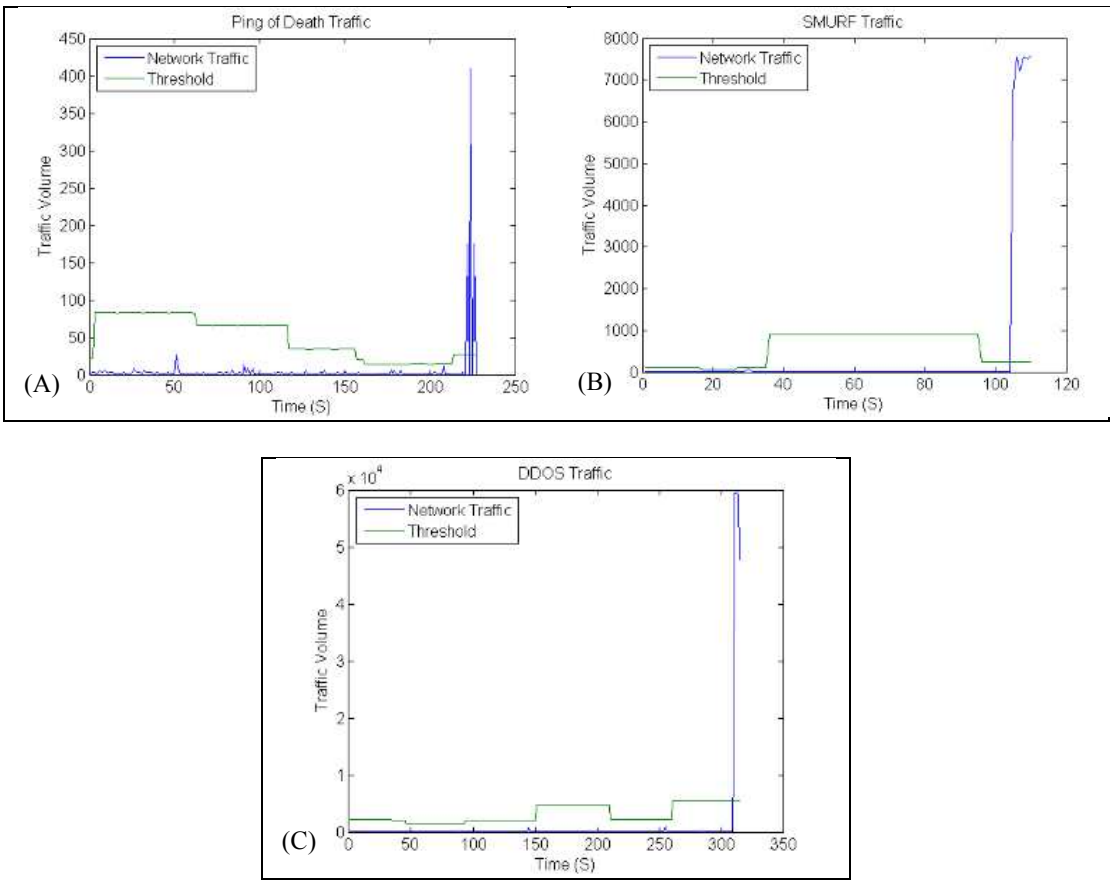

Figure 3. Behavior of network traffic during POD (A), SMURF (B) DDOS (C) attacks.

As mentioned before when the network traffic passes the threshold the system will flag that specific time slot as suspicious and the traffic in that time slot will undergo for clustering. Figure 4, (A_1, A_2, and A_3) shows the self-training phase of attacks such as POD, SMURF and DDOS. As DBSCAN needs two parameters "ε" and "*minPts*" (maximum radius of the neighborhood and minimum number of samples required to form a cluster), it sets "ε" as average Euclidian distance between all of the points in the dataset and "*minPts*" equal to 10% of the traffic sample. In Self-Training phase the clustering engine cluster the 5 min attack free time slot (the time slot before the suspicious behavior) to find out more accurate "ε". Then it will obtain the minimum distance between noises and clustered points as "φ". Based on our experimental results, sum of "φ" and "ε" will create more accurate and acceptable radius of the neighborhood for the DBSCAN during attack detection. Figure 4, (B_1, B_2, and B_3) shows the detection phase of POD, SMURF and DDOS attacks in which the noises are pointed by arrows in the diagram showing the attackers or the victim of the attacks. For instance during 1 to N network attacks such as POD as the intruder creates high amount of network flows the clustering engine consider this behavior as anomaly or non-self and in N to 1 network attacks such as SMURF or DDOS as the victims are flooded by huge number of requests the clustering engine considers this behavior as noise or non-self.

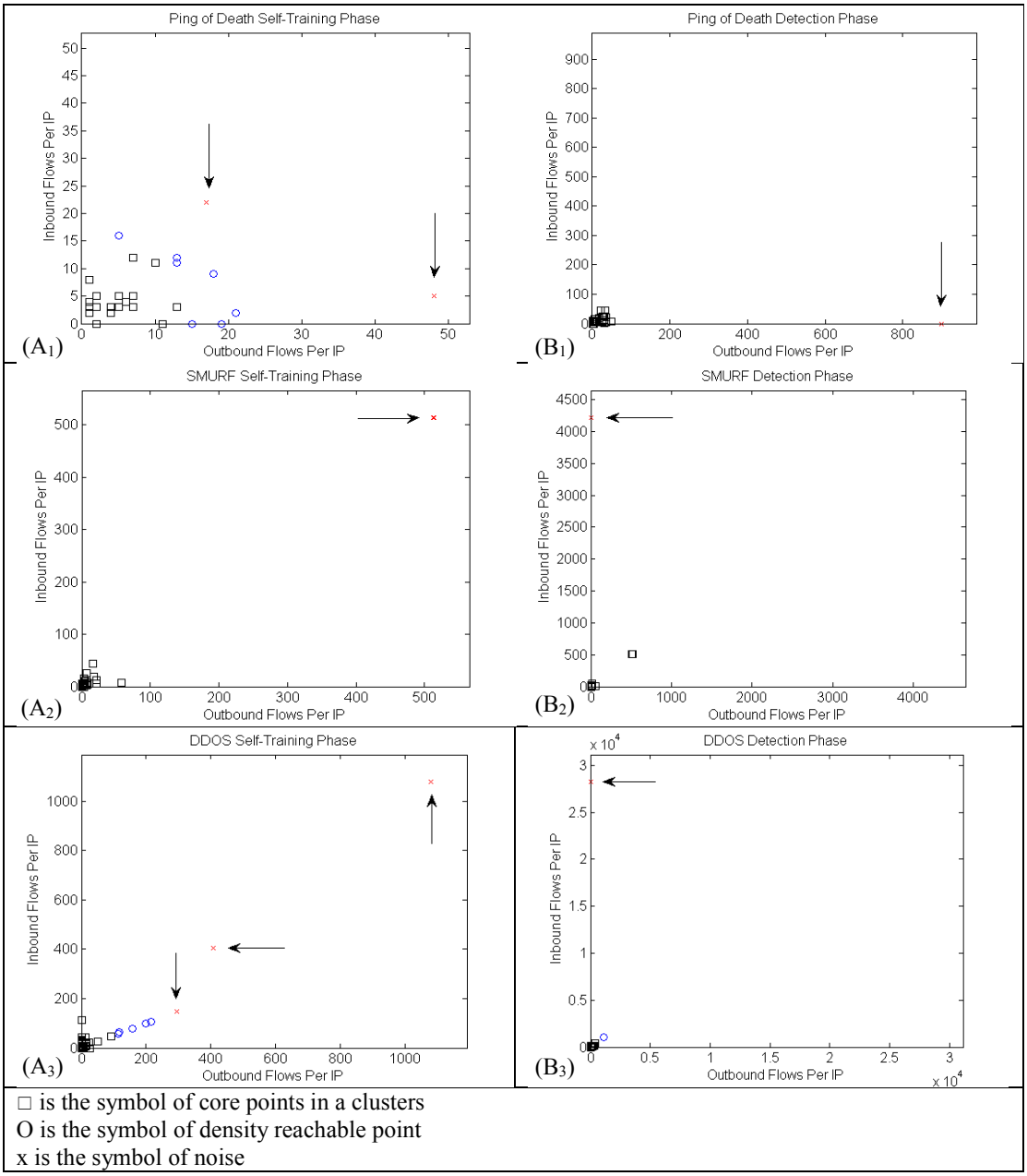

Figure 4. Detection of network attacks by the clustering algorithm

In order to test the efficiency of the AIS engine and thus the proposed model, in our experiment the fitness value of "rc" for R-Contiguous matching bit algorithm is set to 13 and threshold of "T_i" is set to 3. Moreover, by testing the genetic algorithm for generation of memory cells, in different conditions, the probabilities of genetic operations of Crossover, Mutation, and Reproduction have been fixed to 30%, 40% and 30% respectively. The system is tested in both centralized and distributed mode. Figure 5 compares the self-improvement rate of AIS based IDS in central and distributed modes. According to this diagram, the self-improvement rate in distributed mode is better than centralized mode and it reaches to its stable maximum amount after only 6 rounds, while this happens after 10 rounds in centralized mode. This is because of dynamic distribution and synchronization of newly generated memory cells in each host to others.

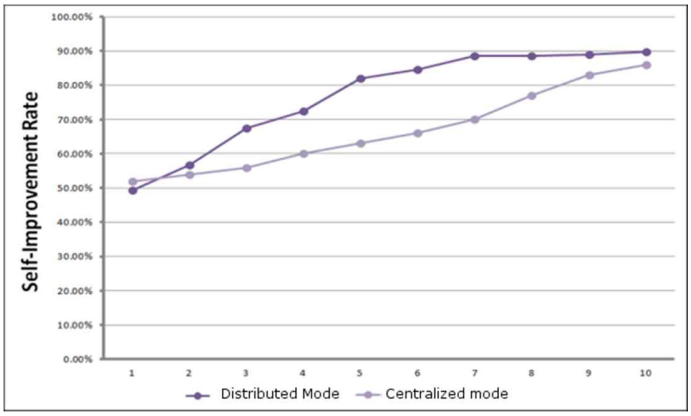

Figure 5. Comparison of self-improvement rate in distributed and centralized mode.

8. Conclusion and Future Work

In this paper, we presented a novel architecture for an intrusion detection system based on the artificial immune system. We proposed innate immunity using unsupervised machine learning methods. According to our primary experiments we conclude that among other methods, DBSCAN clustering is robust and has the greatest potential for this purpose. In this multi-layered framework, the clustering engine labels the network traffic as self and non-self without previous training or knowledge about network flow profiles, thus acting as the first line of defense in AIS-based IDS and providing innate immunity. We defined a network measurement formula as a dynamic threshold to facilitate the detection of abnormal network behaviors. The output of clustering is used to feed the training data for the adaptive immune system as online and real-time training data. Primary detectors after training are distributed to hosts in the network and provide primary immune response for our IDS. We presented the experimental results of proposed innate immune mechanism using our network measurement formula. We also demonstrated that the distributed structure for this IDS is more efficient than the centralized mode. Suspected intrusions reported from hosts are analyzed and an optimized memory cell detector is generated through a genetic algorithm process. Memory cells are attack specific detectors, which provide a secondary immune response. We defined detector life cycle to update and eliminate weak or inefficient detectors to enhance the performance of detection. Future work will mainly focus on detection of potential Bots and BotMater after DDOS attacks. As there is a specific behavioral structure in communications between the BotMaster and Bots in Botnet attacks, it is possible to cluster the communication between potential Bots (DDOS attackers) to detect the BotMaster.

9. References

[1] L. N. de Castro and J. Timmis, Artificial {I}mmune {S}ystems: A {N}ew {C}omputational {A}pproach. London. UK.: Springer-Verlag, 2002, p. 357.
[2] S. Forrest, S. A. Hofmeyr, and A. Somayaji, “Computer Immunology,” Commun. ACM, vol. 40, no. 10, pp. 88–96, Oct. 1997.
[3] S. A. Hofmeyr and S. A. Forrest, “Architecture for an Artificial Immune System,” Evol. Comput., vol. 8, no. 4, pp. 443–473, Dec. 2000.
[4] S. and G. G. Feixian, “Research of Immunity-based Anomaly Intrusion Detection and Its Application for Security Evaluation of E-government Affair Systems.,” JDCTA Int. J. Digit. Content Technol. its Appl., vol. 6, no. 20, pp. 429 – 437, 2012.
[5] M. Tan, H. Yu, Z. Zhao, Z. Liu, and F. Liu, “An artificial immunity-based proactive defense system,” in Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on, 2007, pp. 2239–2243.
[6] et al. Xishuang, D., “Multi-word-Agent Autonomy Learning Based on Adaptive Immune Theories,” JDCTA Int. J. Digit. Content Technol. its Appl., vol. 7, no. 3, pp. 723–745, 2013.

[7] A. A. Ademokun and D. Dunn-Walters, “Immune Responses: Primary and Secondary,” in eLS, John Wiley & Sons, Ltd, 2001.
[8] P. Casas, J. Mazel, and P. Owezarski, “Unsupervised Network Intrusion Detection Systems: Detecting the Unknown Without Knowledge,” Comput. Commun., vol. 35, no. 7, pp. 772–783, Apr. 2012.
[9] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, “An Effective Unsupervised Network Anomaly Detection Method,” in Proceedings of the International Conference on Advances in Computing, Communications and Informatics, 2012, pp. 533–539.
[10] T. T. T. Nguyen and G. Armitage, “A survey of techniques for internet traffic classification using machine learning,” Commun. Surv. Tutorials, IEEE, vol. 10, no. 4, pp. 56–76, 2008.
[11] P. K. Harmer, P. D. Williams, G. H. Gunsch, and G. B. Lamont, “An artificial immune system architecture for computer security applications,” Evol. Comput. IEEE Trans., vol. 6, no. 3, pp. 252–280, Jun. 2002.
[12] F. Hosseinpour, K. A. Bakar, A. H. Hardoroudi, and N. Kazazi, “Survey on Artificial Immune System As a Bio-inspired Technique for Anomaly Based Intrusion Detection Systems,” in Proceedings of the 2010 International Conference on Intelligent Networking and Collaborative Systems, 2010, pp. 323–324.
[13] J. D. Farmer, N. H. Packard, and A. S. Perelson, “The immune system, adaptation, and machine learning,” Phys. D Nonlinear Phenom., vol. 22, no. 1–3, pp. 187–204, 1986.
[14] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, “Self-nonself discrimination in a computer,” in Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on, 1994, pp. 202–212.
[15] P. Matzinger, “Essay 1: The Danger Model in Its Historical Context,” Scand. J. Immunol., vol. 54, no. 1–2, pp. 4–9, 2001.
[16] P. Matzinger, “The Danger Model: A Renewed Sense of Self,” Science (80-.)., vol. 296, no. 5566, pp. 301–305, 2002.
[17] U. Aickelin, P. Bentley, S. Cayzer, J. Kim, and J. McLeod, “Danger Theory: The Link between AIS and IDS?,” in Artificial Immune Systems, vol. 2787, J. Timmis, P. Bentley, and E. Hart, Eds. Springer Berlin Heidelberg, 2003, pp. 147–155.
[18] D. Dal, S. Abraham, A. Abraham, S. Sanyal, and M. Sanglikar, “Evolution Induced Secondary Immunity: An Artificial Immune System Based Intrusion Detection System,” in Computer Information Systems and Industrial Management Applications, 2008. CISIM ’08. 7th, 2008, pp. 65–70.
[19] Hosseinpour F., Meulenberg A., Ramadass S., Vahdani Amoli P., and Z. Moghaddasi, “Distributed Agent Based Model for Intrusion Detection System Based on Artificial Immune System,” Int. J. Digit. Content Technol. its Appl., vol. 7, pp. 206–214, 2013.
[20] M. Amini, R. Jalili, and H. R. Shahriari, “RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks,” Comput. Secur., vol. 25, no. 6, pp. 459–468, 2006.
[21] T. Stibor, “Foundations of r-contiguous Matching in Negative Selection for Anomaly Detection,” Nat. Comput., vol. 8, no. 3, pp. 613–641, Sep. 2009.