Agent based Parallelized Intrusion Detection System using Ant Colony Optimization

D.P. Jeyepalan
Research Scholar,
School of Computer Science, Engineering and Applications,
Bharathidasan University, Tiruchirappalli,
Tamilnadu, India.

E. Kirubakaran
SSTP (Systems),
Bharat Heavy Electricals Ltd,
Tiruchirappalli, India.

ABSTRACT
Detecting intrusions in a network is one of the major functionalities that cannot be overlooked. Even though the intrusion detection systems in networks tend to perform their best, the other side is always ahead conjuring new attacks every time. Further, detecting an attack earlier or at least as soon as the attack has occurred is the only way to counter it. Detecting it at a later point in time proves to be useless. But the current systems available are not able to live up to the needs of the real time scenario. This paper presents an Ant Colony Optimization based intrusion detection system that uses agents to perform the process of detection, storage and monitoring. The network is not considered as a whole, instead, it is divided into clusters and detection is performed on the nodes within the cluster alone. Hence the workload of the detection system is reduced considerably, providing faster results. Another added advantage is that all the agents can run in parallel, hence parallelized detection becomes possible. Experiments were carried out using multi core CPUs and many core GPUs and the comparison shows that the CPUs shows twice the increase in performance when compared to single core machines, while GPUs show thrice the increase in performance when compared to multi core CPUs.

Keywords
Intrusion detection; Parallelized ACO; Clustering; Cluster Head Selection; Agent based IDS

1. INTRODUCTION
The internet age has increased the complexity and size of the networks to a very large extent. While this proves to be a promising activity, that can provide improvement in all the fields depending on networked technologies, it also contains a dark side that reveals that more information being brought online directly infers that more information become vulnerable and can be exploited using vulnerabilities in the network. Detecting intrusions in a network is one of the major functionalities of any network administration system. The last two decades have seen a numerous growth in networking technologies, attacks, as well as technologies used to counteract these attacks. Many strategies and methods have been developed for counteracting intrusions, but new types of intrusions can be found on raise every day.

A major problem that occurs in the detection of intrusions is the probability of an intrusion detection system to generate false positives. False positives refer to marking legitimate transmissions as intrusions and blocking them. The expectation of a good intrusion detection system is not only to detect intrusions appropriately; it is also expected to provide the least false positive rate.

A probabilistic agent based intrusion detection mechanism is proposed in [1]. It uses cooperative agent based architecture for the detection process. Each agent is assigned a task of its own and hence provides a distributed processing structure. Further, the agents are capable of self-evidential update which provides intrusion detection on a continuous scale. Agent communication occurs in XML syntax. They commonly exchange registration requests and information requests. The downsides of this approach is that too many data transfers tend to happen, hence cost of data transfer increases. Reliability and security is achieved by incorporating commercially available encryption techniques.

A summary of the existing anomaly based network intrusion detection techniques is discussed in [2]. It also provides the available systems that perform the required task, along with the major challenges constituting the process. A classification of the techniques related to their behavioral nature is also presented in detail.

An agent based intrusion detection system is presented in [3]. It has been proposed to detect malfunctions, abnormal traffic, faults, intrusions, deviations, and also claims to provide recommendations to the user. The advantage of this approach is that it can simultaneously monitor the network activities at multiple levels, and hence can effectively provide correlation among the deviated values and security violations. The agents are implemented using the Cougaar framework, with each node containing four agents at its disposal. CIDS (Cougaar based Intrusion Detection System) being a modular design, has the flexibility to incorporate new detection, action and decision rules. It contains a swing based GUI, hence it can also act as a monitoring agent. The method claims to detect probing attacks and DoS attacks in an effective manner.

Using an accurate intrusion detection method is not just sufficient for building an effective intrusion detection system. The ID method to be used should be able to perform faster such that the results returned by an IDS is actually usable. [4] presents an attack graph [5,6] based prediction system that performs the process in linear time with quadratic space requirements. It presents a queue graph (QG) approach that uses the latest alerts for processing. Hence all the past alerts need not be examined, which reduces the amount of processing to a large extent. Further, it also helps correlate alerts that are arbitrarily far away, hence can help predict intrusions effectively.

A GPU based parallel Intrusion Detection mechanism is described in [15]. It converts commodity GPU hardware into a powerful pattern matching processor. Parallelized packet inspection is carried out, which provides an efficient and also faster method for intrusion detection. A regular expression based IDS, using GPUs has been discussed in [16]. Due to the highly comparing nature of the regular expressions, an

overhead is added to the system due to frequent CPU and memory access. Usage of GPUs is found to reduce this overhead by performing the tasks in parallel. The system claims to produce an increased speed up of about 48 times when compared to regular systems.

A GPU based IDS that provides active protection along with improved network security is presented in [17]. The IDS named Suricata also helps in performing effective traffic analysis. It uses the CUDA architecture (GeFORCE GTX 260) for implementation and uses rule based computations. A similar method is proposed in [18], which uses a 12 core machine with 2 GPUs. This method named the Kargus claims to exploit the full potential of commodity hardware. It is a batch processing system, that claims to adapt its resource usage depending on the input rate, hence is said to be power saving.

The remainder of this paper is structured as follows; section II provides the system architecture, section III discusses in detail the intrusion detection approach proposed in this paper, section IV presents the results and discussion, and section V concludes the study.

2. SYSTEM ARCHITECTURE

The process of systematizing the distributed intrusion detection system is performed in two phases. The initial phase deals with the deployment of agents in the network and the second phase deals with the creation of an intrusion detection system using the process of Ant Colony Optimization (ACO).

CUDA architecture has been used for the implementation of ACO. CUDA (Compute Unified Device Architecture), is a parallel computing platform and programming model developed by NVIDIA. CUDA enables the GPUs to be used for general purpose computations. GPUs comprising of parallel architecture, can effectively execute multiple threads (thousands of threads) at the same time in comparison to a very few threads in the case of CPUs. Several publications have reported an order of magnitude increase in performance (from 10x to 300x).

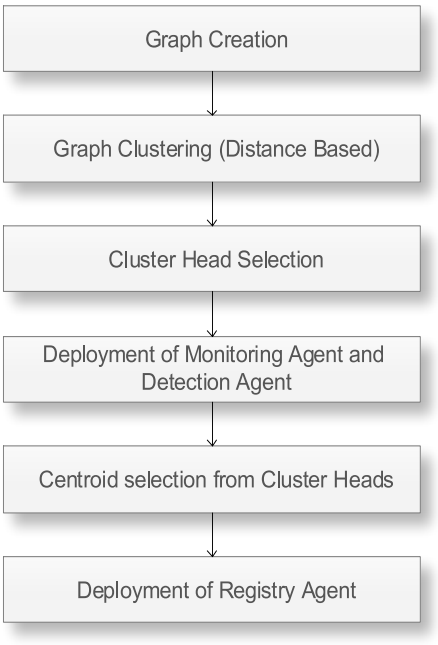

Fig 1: Agent Deployment

Fig 1 presents the initial phase of the intrusion detection system. The three agents used in the process are Registry Agent (RA), Monitoring Agent (MA) and the Detection Agent (DA). The network is clustered and appropriate points for deploying these agents are determined. The intrusion detection mechanism is made to run on the Detection Agents, which help in the process of Intrusion Detection.

3. AGENT BASED PARALLELIZED INTRUSION DETECTION SYSTEM USING ANT COLONY OPTIMIZATION

Intrusion Detection is one of the vital functionalities of any network. Moreover, this is the major component in maintaining the stability of a network. Thetwo phase process of intrusion detection is carried out in the form of a deployment process and the core intrusion detection process.

The deployment process deals with selecting the appropriate nodes for the agents. The process of Clustering is carried out and cluster head detection is performed on the network [7].Clustering is performed using Ant Colony Clustering algorithm [12], and Cluster head selection is performed on the basis of Game Theory. After the determination of cluster heads, the agent deployment phase commences.

The three agents that are to be deployed in the system are:

- **Monitoring Agent**
  The monitoring agent records all the activities that take place in the system. It maintains the log details of all the transmissions that take place in the network.
- **Detection Agent**
  The process of intrusion detection is carried out by the detection agent. The Ant Colony Optimization algorithm is used for performing the detection process.
- **Registry Agent**
  All the intrusions detected by the system are recorded in the registry agent. This repository maintains all the intrusion signatures to facilitate the detection of misuse based intrusions.

The cluster heads identified from the clustering process serves as the destinations for the Detection and the Monitoring Agents. The Monitoring Agent keeps track of the traffic flowing through the corresponding cluster. Hence the overload that is built on a single system to monitor the entire network is reduced. Due to this advantage the system obtains over the conventional methods, it facilitates the intrusion detection agent to also run on the same system. The cluster heads that have been determined earlier are used as points, from which the centroid is calculated. The centroid cluster head acts as the Registry Agent. The Registry Agent usually has the most system accesses. Hence the functionality for the centroid is restricted to be the Registry Agent. Due to the centrality of the location of the Registry Agent, the length of transmission is almost the same when considering from any cluster head. This provides a good amount of traffic reduction, and hence in turn will reduce the bandwidth consumption. When considering a network in general, anomalies occupy a very small portion of the total transmissions. Hence it can be justified that anomalies be stored in one centralized system would be sufficient, while legitimate transactions should be distributed in order to reduce the load.

The Detection Agent [13]is constructed using the Ant Colony Optimization algorithm. The Ant System is a collection of algorithms that perform the process of Ant Colony Optimization. For the process of Intrusion Detection, we consider the Ant Cycle variant as the base algorithm.

The graph is constructed by considering every individual transmission as a node. Two graphs are considered here. A graph with normal transactions, for the process of anomaly based detection and a graph with anomalous transactions for the process of misuse based detection. The Ant Cycle algorithm is subject to a few modifications to accommodate it to the process of Intrusion Detection. Number of ants to be used for the detection process depends upon the importance of the system. All the ants initially occupy a node representing the current transmission (not any random node as in the case of normal ACO). Since our graph contains both normal and anomalous transmissions as nodes, the destination route from the current transmission will automatically indicate whether it is normal or anomalous [14].

Algorithm:

1. *Graph Creation using the transmission data in Monitoring Agent and anomaly data in Registry Agent*
2. *Ant deployment in current transaction*
3. *Distance assignment using the transmission properties*
4. *Find node j to move to with the probability* P_{ij}
5. *Find the distance between i and j*
6. *Consider the distances returned by the ants and find the shortest distance and its corresponding node*
7. *Repeat 4,5 and 6 in parallel for every deployed ant*
8. *Repeat 4,5,6 and 7 until the stopping criterion is met*
9. *Check the containment clause of the node corresponding to the shortest distance*
10. *If it is from the anomalous clause, then the current transaction is considered to be an anomaly else it is considered to be a normal transaction.*

The initial step that is to be performed for an Ant Colony Optimization algorithm [10] is the creation of the graph. After completing every transmission, the nodes in the network writes the transmission details in the log files.The data from log files are considered as the base for creating graphs. By analyzing the nature of this problem, it can be seen that the number of nodes in the graph keeps increasing, since every transaction is to be added to the graph as a node. Due to the huge increase of data, the monitoring agent contains logs only from the current cluster, which makes the detection process easier. Hence the process of detection is carried out only within the same cluster, which makes the detection process parallel (all DA's can run simultaneously) and faster. The detection agent runs after every transaction in the cluster. Due to its parallelized nature, the detection process is particularly fast and hence it can cater to the requirements of the real time processing.

The Ant Cycle algorithm is one of the variants of the Ant System. In general, ACO considers the pheromone concentration τ and the evaporation rate ρ as the basic parameters for optimization.

The probability of an ant to travel to a node is given by,

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{l\in J_k(i)}\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}, if & j\in J_k(i) \\ 0, & j\notin J_k(i) \end{cases} \quad (1)$$

Here*i*and *j* refer to the nodes, P_{ij}^{k}(t) is the probability of an ant (k) to travel from i to j,$\tau_{ij}(t)$ is the pheromone intensity between i and j at time t, $\eta_{ij}(t)$ is the pheromone visibility between i and j, α and β are the importance of pheromone intensity and visibility respectively and $J_k(i)$ is the neighborhood set of node i for the ant k.

Pheromone updates are performed based on,

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k}, & if(i,j)\epsilon tour \\ 0, otherwise \end{cases} \quad (2)$$

Here,

$\Delta_{\tau_{ij}^{k}}$ is the amount of pheromone updated between i and j, Q is the constant and $L_k$is the total path length covered by ant k.

Pheromone updates are given by,

$$\tau_{ij}^{k} = \tau_{ij}^{k} + \Delta\tau_{ij}^{k} \quad (3)$$

and pheromone decay is given by,

$$\tau_{ij} = (1-\rho) \quad (4)$$

Here

τ_{ij} is the pheromone intensity between i and j and ρ is the evaporation parameter of the graph.

ACO, being an intrinsically parallel metaheuristic algorithm, it has the ability to perform more efficiently if implemented in parallel so a parallelized version of ACO [8,9] is used for performing the optimization process. After the resultant node is determined by the ACO[11], its category is analyzed. If it belongs to the normal transmissions from the monitoring agent, then the current transmission is considered to be a legitimate transmission and is logged into the monitoring agent, and if it is found to be a transmission from the registry agent, then the current transmission is found to be anomalous and the protection mechanisms are triggered and it is written on to the registry agent.

4. RESULTS AND DISCUSSION

The intrusion detection system using parallelized ACO was constructed using C#.NET and CUDA C. Experiments were carried out on a core i7 quad core machine running at 2 GHz with 4GB RAM. GPU setup includes a NVIDIA GeForce GT740M GPU with 2 SM units containing 384 shader units in total with 2 GB DDR3 Graphics Memory and a compute capability of 3.0 based on Kepler Architecture.

The time taken for detecting intrusions using CPUs (4 core multithreaded) was calculated and is shown in Figure 2. The time varies from 411 ms in the highest point to 76 ms as the

minimum. The average time taken for the processing is found to be 159 ms. Hence it can be seen that the time taken for processing is very minimum and the system is capable of detecting attacks in real time.

Time Taken for decision making using GPUs is shown in Figure 4. The minimum time taken for the detection process was found to be 25ms, while the maximum time of 137ms was recorded. The average time taken for the detection process was found to be 52.46ms, which is approximately three times speedup when compared to the CPU based process.

The speedup achieved using GPUs is found to be higher, due to the massively parallel nature in comparison to the restricted vectorization like parallelism in multi-threaded CPUs and also due to the fact that the memory in the case of GPU is completely programmable(helps in overcoming the memory wall problem) and optimized for fast access in comparison to the CPU which suffers from cache miss to a considerable extent resulting in a memory wall sooner in comparison to a GPU based implementation.

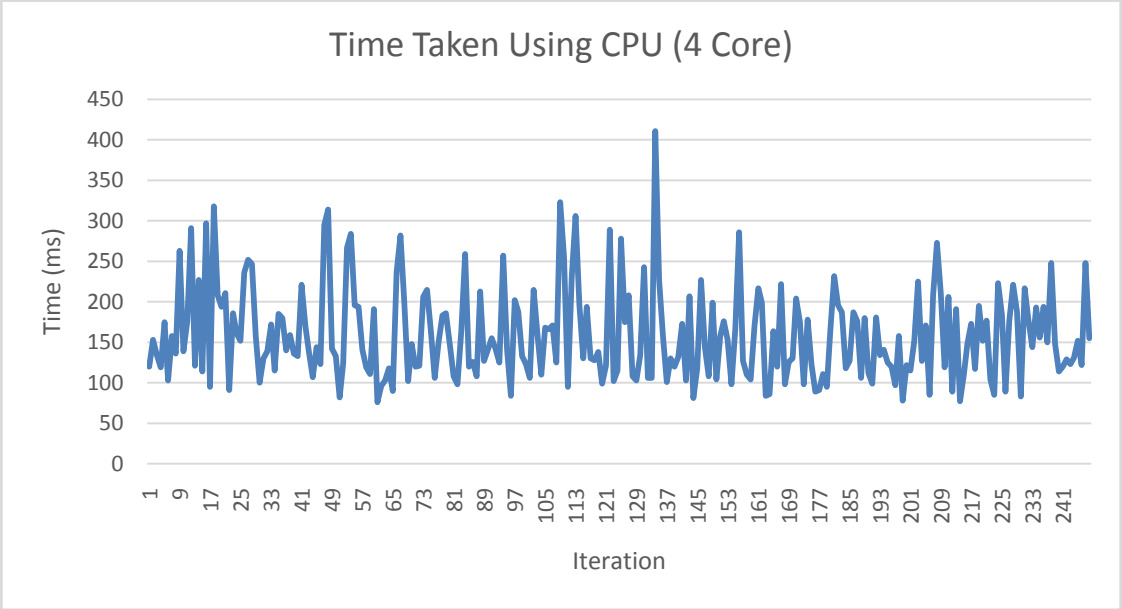

Fig 2: Time Taken for decision making (CPU-4 cores multithreaded)

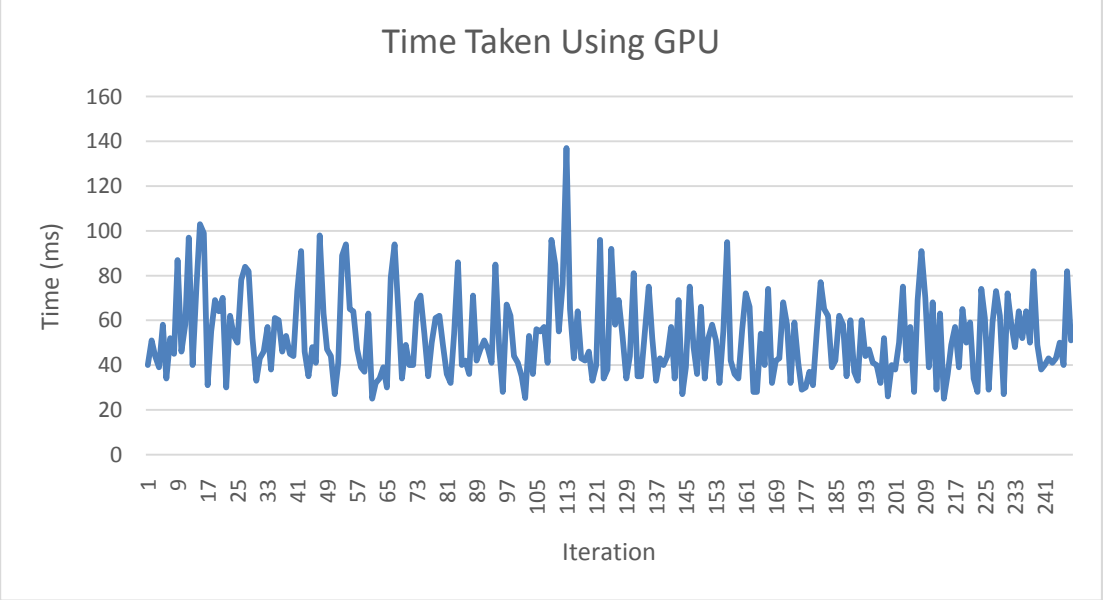

Fig 3: Time Taken for Decision Making (GPU)

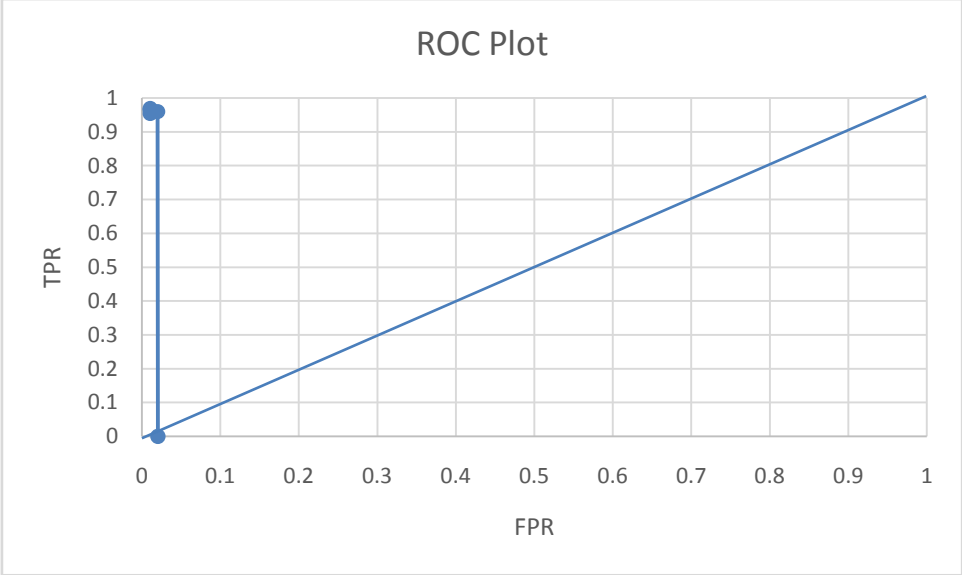

Fig 4: ROC Plot

The ROC plot for the current system is shown in Figure 4. The prediction accuracy of the system is denoted by the curve's movement towards the upper left corner, i.e. towards (0,1). It can be observed that the points in the curve moves towards (0,1) which proves that our prediction system works towards achieving maximum accuracy. The accuracy of the system is found to be 0.976285.

5. CONCLUSION

The result shows that the distributed intrusion detection system not only provides intrusion detection in real time, but also with low false positive rates. Due to the distributed nature of the system, the amount of processing is hugely reduced. One downside of this approach is that it cannot be used in dynamic networks as such. Methodologies for periodically checking the status and members of the network, to determine the new cluster heads can be included in case of usage with dynamic networks. The registry agent maintains the signatures of the anomalies; hence it is bound to be shared by all the other nodes. Even though this tends to increase the bandwidth consumption, the numbers of malicious entries are very few when compared to the number of legitimate entries, and they do not occur very often, so the transmission from registry agent need not occur often. Other cluster heads can be made to maintain the anomaly details and perform a few updating procedures in case of a new intrusion. The process of maintaining consistency in such an approach can be performed in future. Further extension of this approach is to improve the system to fit into the Hadoop environment so as to make use of the massively parallel capabilities of the Hadoop Platform. Since this method already uses a distributed storage and processing methodology, the transition to Hadoop can be easily performed.

6. REFERENCES

[1] VaibhavGowadia ,CsillaFarkas , Marco Valtorta. 2005 Paid: A probabilistic agent-based intrusion detection system. Computers & Security 24(7):529–545.

[2] Garcia-Teodoro, Pedro, et al. 2009 Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers& security 28.1: 18-28.

[3] Dasgupta, D., et al. 2005. CIDS: An agent-based intrusion detection system. Computers & Security 24.5: 387-398.

[4] Wang, Lingyu, Anyi Liu, and SushilJajodia. 2006 Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts. Computer communications 29.15: 2917-2933.

[5] P. Ammann, D. Wijesekera, S. Kaushik. 2002 Scalable, graph-basednetwork vulnerability analysis, in: Proceedings of the 9th ACMConference on Computer and Communications Security (CCS'02), pp. 217–224.

[6] O. Sheyner, J. Haines, S. Jha, R. Lippmann, J.M. Wing. 2002. Automatedgeneration and analysis of attack graphs, in: Proceedings of the 2002IEEE Symposium on Security and Privacy (S& P'02), pp.273–284.

[7] D. P. Jeyepalan, E. Kirubakaran. 2014. A Co-operative Game Theoretic Approach to Improve the Intrusion Detection System in a Network using Ant Colony Clustering. International Journal of Computer Applications, Volume 87 - Number 14.

[8] Foschini, Luca, et al. 2008. A parallel architecture for stateful, high-speed intrusion detection. Information Systems Security. Springer Berlin Heidelberg,203-220.

[9] Vasiliadis, Giorgos, MichalisPolychronakis, and Sotiris Ioannidis. 2011. MIDeA: a multi-parallel intrusion detection architecture. Proceedings of the 18th ACM conference on Computer and communications security. ACM.

[10] Abadeh, Mohammad Saniee, JafarHabibi, and EmadSoroush. 2008. "Induction of Fuzzy Classification systems via evolutionary ACO-based algorithms."computer 35: 37.

[11] Feng, Wenying, et al. 2014. Mining network data for intrusion detection through combining SVMs with ant colony networks." Future Generation Computer Systems 37: 127-140.

[12] Kolias, Constantinos, GeorgiosKambourakis, and M. Maragoudakis. 2011. Swarm intelligence in intrusion detection: A survey. Computers& security 30.8: 625-642.

[13] Catania, Carlos A., and Carlos GarcíaGarino. 2012. Automatic network intrusion detection: Current techniques and open issues. Computers & Electrical Engineering 38.5: 1062-1072.

[14] Shamshirband, Shahaboddin, et al. 2013. An appraisal and design of a multi-agent system based cooperative wireless intrusion detection computational intelligence technique. Engineering Applications of Artificial Intelligence 26.9: 2105-2127.

[15] Huang, Nen-Fu, et al. "A GPU-based multiple-pattern matching algorithm for network intrusion detection systems." Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on. IEEE, 2008.

[16] Vasiliadis, Giorgos, et al. "Regular expression matching on graphics hardware for intrusion detection." Recent Advances in Intrusion Detection. Springer Berlin Heidelberg, 2009.

[17] Vokorokos, Liberios, Anton Baláž, and BranislavMadoš. "Intrusion detection architecture utilizing graphics processors." ActaInformaticaPragensia 1.1 (2013): 50-59.

[18] Jamshed, Muhammad Asim, et al. "Kargus: a highly-scalable software-based intrusion detection system." Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012.