DISTRIBUTED INTRUSION DETECTION SYSTEM FOR COMPUTATIONAL GRIDS

M. F. Tolba M. S. Abdel-Wahab I. A. Taha A. M. Al-Shishtawy

alshishtawy@yahoo.com, Tel. +20105690130
Scientific Computing Department
Faculty of Computer and Information Sciences
Ain Shams University
Cairo, Egypt

Abstract:

Applying intrusion detection to the fast growing computational Grid environments improves the security which is considered to be the heart of this new field. Flexible cooperative distributed intrusion detection architecture is introduced that suits and benefits from the underlying computational Grid environment. The proposed architecture was tested using homogeneous distributed intrusion detection servers that use learning vector quantization neural network to detect the intrusion if occurred. The paper discusses the different parameters that may affect the proposed intrusion detection system showing and explaining their effects on the overall system performance.

Keywords:

Computational Grids, Grid Security Architecture, Intrusion Detection.

1. Introduction

Grid Computing is a new approach for computing and problem solving. It has been proposed in the mid 90's and still under research, aiming to achieve seamless access to computational power and resources, similar in simplicity to the access to electricity through the electrical power grid [5]. Grid Computing was defined in [11] as "coordinated resource sharing and problem solving in a dynamic, multi-institutional virtual organizations". Grid Computing has special characteristics, including heterogeneity, scalability, and dynamicity or adaptability [14]. It also has special requirements such as to coordinate resources that are not subject to centralized control, use standard open general-purpose protocols and interfaces, and deliver nontrivial qualities of service [6]. These special characteristics and requirements introduced new challenges to researchers trying to design the architecture, infrastructure, and basic tools and services necessary to construct computational Grid environments. These challenges can be classified in four main research fields [20][21]: Resource Management, Data Management, Information Services, and Security.

Security is one of the most important features that must exist to enable the creation of Grid environments that couple multiple locally administrated sites and resources to solve real scientific and/or business applications. The special characteristics and requirements of Grid environments have introduced unique security requirements that did not exist in old security mechanisms [10]. Most of the available attempts to secure grid environments are based on public key infrastructure focusing on authentication and access control. They try to satisfy, in addition to normal security requirements, special requirements for Grid environments including single sign on, interoperability with local security solutions, exportability, and support for multiple implementations [10].

Intrusion detection is considered as a second line of defense. It is very important because the current grid security mechanisms can be penetrated and also does not provide protection from insiders. Intrusion detection systems are based on the assumption that normal use of the system is different from malicious use [17]. Due to the special characteristics of Computational Grids, detecting such difference in behavior in a grid intrusion detection system imposed some new unique requirements that did not exist in traditional intrusion detection systems.

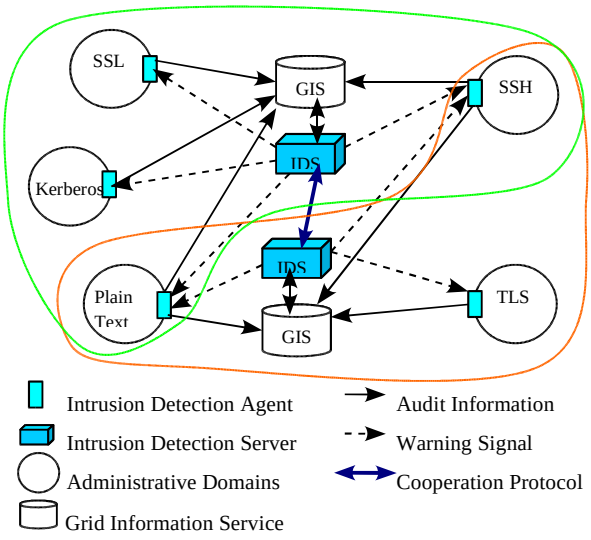

Figure 1. Proposed Grid intrusion detection

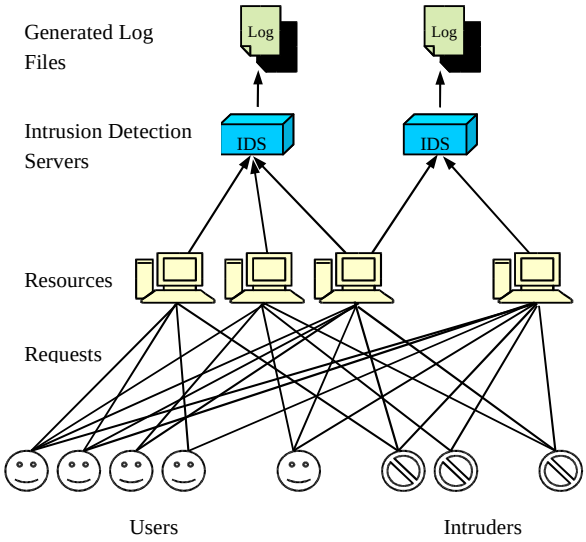

Figure 2. The simulated Grid and data gathering modules.

Traditional intrusion detection systems such as [13] depend on a centralized server that is capable of monitoring and analyzing the entire system to detect intruders. Centralized intrusion detection architectures are not suitable for Computational Grids because of its poor scalability and centralized control among others. Also because of the fact that the grid consists of resources controlled by different administrative domains it is not possible to find a single intrusion detection server that all these administrative domains can trust, agree to use and depend on.

A Grid Intrusion Detection Architecture (GIDA) was introduced in [16]. Section 2 will present the proposed GIDA and discuss its compatibility with Grid environments. Section 3 will present a possible implementation of the GIDA that will be used in testing. Results and major parameters effects on the system are discussed in Section 4. Conclusions and Future work are discussed in section 5.

2. The Proposed Grid Intrusion Detection Architecture (GIDA)

Distributed intrusion detection systems such as [12] are more suitable for Computational Grids. They have enhanced scalability by distributing some of the system components, such as the modules responsible for gathering information about the system while keeping the module responsible for analysis and detection of intruders centralized or in some system taking a hierarchical form. Distributed systems, although enhanced, are still not sufficient for Computational Grids. The components which are left centralized or components near the top of a hierarchy forms a performance bottle neck, a single point of failure and force centralized control and administration. In Grid environments, complex trust relationships must be addressed by intrusion detection systems and all the components must not be subject to a centralized control.

GIDA was designed with all these problems in mind. It is built on top of the Grid Security Infrastructure GIS [10] which provides a uniform security infrastructure for Computational Grids and inter-operates with the diverse intra-domain security solutions. As shown in (Figure 1) GIDA has two main parts. The first is the data gathering module, called the Intrusion Detection Agent (IDA), which is responsible for gathering information about the users and resources. The second part which is called the Intrusion Detection Server (IDS), consists of two modules. The first module is responsible for analyzing the gathered information while the second cooperates with other IDSs to detect intruders. Both parts, IDA and IDS, are distributed and not subject to centralized control as shown in (Figure 1).

As stated above the Computational Grid consists of resources owned by different administrative domains. This is represented by the circles in (Figure 1). Each administrative domain will have an intrusion detection agent responsible for gathering data which is specific to this administrative domain and summarizing these data and converting them to a standard format. In other words this will deal with the heterogeneity of Computational Grids, while summarizing the gathered data will reduce the consumed network bandwidth.

Each intrusion detection agent (IDA) will register with one or more intrusion detection server (IDS) this will increase the reliability, robustness and adaptability of the system. In the case of the failure of one IDS the

administrative domain resources can still be protected against intruders if its IDA is registered with other IDSs. Of course there is a trade off between this increased reliability and robustness and an increased bandwidth consumption and time needed by the IDS to analyze the data because of this replication.

Next the gathered information will be transferred from each intrusion detection agent IDA to all registered intrusion detection servers IDS. The Grid Information Service (GIS) - which is a major component of most Computational Grids - can be used to store this information. If the GIS is unavailable or inapplicable a special database can be implemented in the IDS to store the gathered information for analysis.

The intrusion detection servers will analyze the gathered information and try to detect intruders. These IDSs need not be homogeneous. Each IDS can use a different approach to analyze the gathered data such as anomaly or misuse detection based on neural network, statistical, data mining, or other techniques of intrusion detection. The key here is to use a standard, open, general-purpose protocols between the IDSs that allow them to cooperate and work together. This allows site administrators to choose among different IDSs according to their QoS.

When an IDS detects an intruder it should warn the other IDSs which in turn will signal the registered IDAs that will worn the local security to take an appropriate action.

The administrative domains can have local intrusion detection system that detects local intruders. This local intrusion detection system can cooperate with GIDA to help finding the intruders. The architecture represented in this section is an extensible and open architecture that can be implemented in various ways. It meets the characters and requirements of Computational Grids as discussed above and shown in (Table 1).

Table 1. GIDA compatibility with computational grid characteristic and requirements.

Characteristic or Requirement	GIDA Compatibility
Heterogeneity	IDA deals with heterogeneity
Scalability	All components are distributed
Dynamicity or adaptability	Registration with multiple IDSs
No centralized control	Decision is made through cooperation between IDSs
Standard protocols	Build on top of GSI and Grid protocols
Nontrivial QoS	Different ID algorithms and trust relationships

3. An Implementation of GIDA

The Grid Intrusion Detection Architecture can be divided into three modules: The data gathering module, the analyzing and detection module, and the cooperation module. For the purpose of validating and testing the Grid Intrusion Detection Architecture the data gathering module was simulated to simplify the testing process. The other two modules were implemented then tested using the data generated from the simulation. This section discusses these modules in more details.

3.1. The Data Gathering Module

Computer simulation has always been used as a cost effective solution for the evaluation, testing, and proving the effectiveness of new architectures and models before implementing them in real world applications. Simulation also allows researchers to perform experiments repetitively using different combinations and arrangements in a controlled environment to find the most optimum solution in an effective way that would otherwise be both costly and time consuming or even impossible.

Researchers in the field of Computational Grids face many problems in their research because of the special characteristics of Computational Grids. Most of the researchers do not have access to real Computational Grids or testbeds such as [1][18] to perform their experiments. This is due to the high cost, technical and organizational challenges needed to build a real Computational Grid. Even those who have access to real Computational Grids face

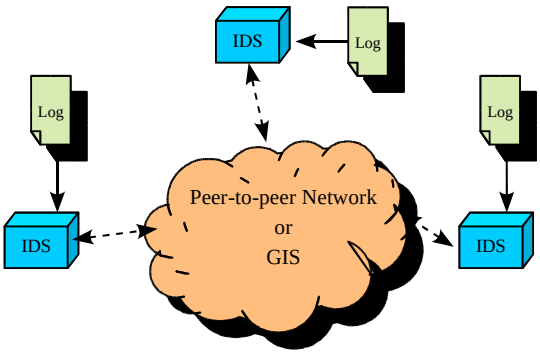

Figure 3. The analysis and detection module consume the simulated data, then cooperate through the cooperation module.

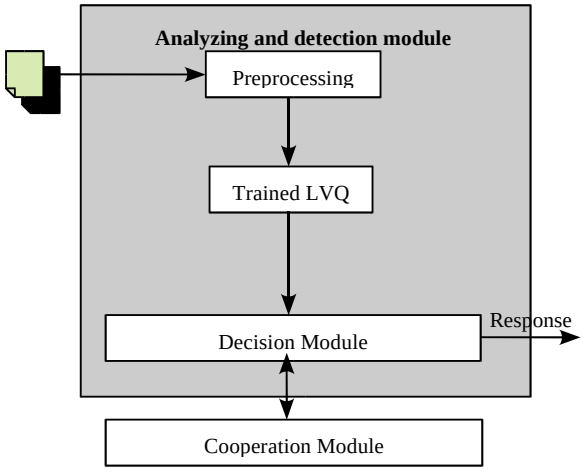

Figure 4. The analysis and detection module.

problems. Computational Grids contain expensive resources such as super computers, large clusters, other expensive devices such as electronic telescope, and so on. Dedicating a portion of these resources' time to researchers to perform their experiments increases research costs and in many cases is infeasible and not applicable.

It is also very difficult to coordinate and control the experiment and gather information about it. This is due to the large size of Computational Grids and the fact that both the resources and users are geographically distributed and are owned by different administrative domains which makes the coordination between them very complex, creating a controlled environment very difficult because of their dynamic nature, and repeating the experiment and testing different combinations and different resource arrangements and scenarios with varying specifications and loads considered impossible. Testing the scalability is another problem which is limited by the size of the Computational Grid available to the researchers.

Because of the above problems most of the researchers have turned to simulate Computational Grids. Tools for simulating Computational Grids have been developed and used in research including for example: GridSim [15], SimGrid [7], and MicroGrid [9]. Researchers use this simulated Computational Grid to test their algorithms, models, and architectures. After they are well established and tested, they are implemented on real Computational Grids and retested only in the final phase. This reduces the time, cost, effort, and accelerates the research and give better results.

Unfortunately most of the available Grid simulation tools are designed to solve problems related to resource management and scheduling ignoring security related requirements such as authentication, authorization, users behavior, and managing trust relationships between different administrative domains. For these reasons a new grid simulation toolkit was developed that addresses security requirements to be used to test the proposed Grid Intrusion Detection Architecture.

The simulation environment simulate users, resources, and registration with IDSs. This allow us to perform the required experiments. Each experiment will generate a dataset consisting of one or more log file as shown in (Figure 2). These datasets are then used to test the analyzing module and the cooperation module of the IDSs.

3.2. The Analysis and Detection Module

This module will analyze the data generated from the simulation, taken advantage of results gained through cooperation, with the goal of detecting intruders trying to compromise and misuse a Computational Grid as shown in (Figure 3).

Intrusion detection systems try to detect, using different mechanisms and approaches, the difference in behavior caused by an intruder and take appropriate action to stop the intruder. Intrusion detection techniques can be classified (Table 2) either according to the source of the data used for the analysis into network based and host based intrusion detection systems [4], or according to the approach taken to analyze the data into misuse detection and anomaly detection [8].

Network intrusion detection systems get their data by installing a device on the network capable of monitoring all network traffic and passed packets. They rely on raw network packets in their analysis. On the other hand host based

Table 2. Different approaches to intrusion detection.

	Misuse	*Anomaly*
Network Based	1	2
Host Based	3	4

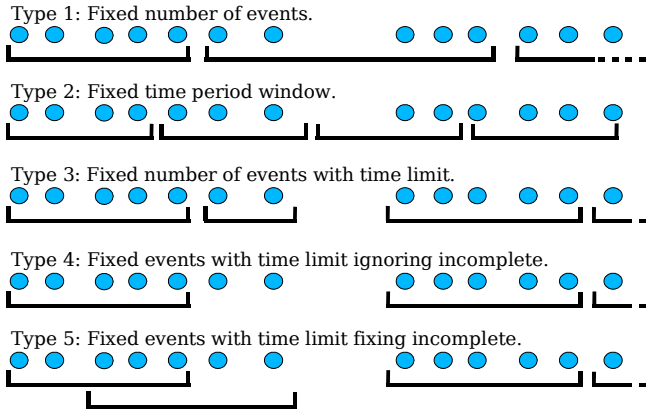

Figure 5. Different Possible Types of Window.

intrusion detection systems use log files created on each host, containing all the operations performed on the host, as the data source. In the context of Computational Grids the network intrusion detection has many disadvantages and problems including:

- It is impossible to have a device installed on the grid capable of monitoring all the passing packets. Even if this device is distributed, moving the raw network packets to the IDS is not efficient so it must be preprocessed and summarized at each administrative domain before being sent to the IDS. This may add undesired overheads and complications to the systems.
- Because of security requirements in the grid most of the raw packets used are encrypted and this cause problems in network based intrusion detection.
- Analysis at a low level such as raw network packets makes higher level information, such as the global name of the user, not available or hard to discover.
- Network based intrusion detection systems analyze the raw network packets to guess what is the user is trying to do. While this information is already available in log files.

For these reasons the presented GIDA implementation is based on host based intrusion detection. The data gathering module is responsible for gathering the data from the log files on each host (or administrative domain) and transferring it to the IDS. Because of these reasons areas labeled (1) and (2) in (Table 2) that use network based approach will not be used.

Misuse detection technique search the gathered data for patterns and signatures of well known attack types stored in a knowledge base, on the other hand anomaly detection technique tries to identify events that appear to be anomalous with respect to normal system behavior [2]. The Grid is still under research and no signatures of known attacks is available to implement miss use detection techniques and so areas labeled (1) and (3) in (Table 2) are excluded.

From the previous discussion only host based anomaly detection intrusion detection technique is currently suitable for Computational Grids (area (4) in Table 2) so it was used to implement the Grid Intrusion Detection Architecture as described in this paper.

Because the GIDA is an open architecture, the analyzing and detection module can be implemented using various techniques. Such as neural networks, statistical analysis, data mining, and so on. It is possible also to be implemented using different technique in different IDSs in the same Computational Grid to increase the QoS.

The implementation described in this paper employs anomaly detection using neural networks on all the IDSs. The neural network used is Learning Vector Quantization (LVQ) [19]. The LVQ was chosen because it dose not require anomalous records in the training data, and because the classes and their labels (global user name) are known. The preprocessing module is responsible for converting attributes in the log file to a format suitable for the neural network. The decision module will analyze the LVQ result then, with information from the cooperation module, will decide wither a user is normal or intruder (Figure 4).

3.3. The Cooperation Module

Each IDS has a scope. This scope is defined by the administrative domains (resources) that chose to register with this IDS as shown in (Figure 1). The analyzing and detection module will make decisions about users based on the data

available in its scope. This will not produce the best results because other important events may occur in the scope of other IDSs. Here arises the important rule of the cooperation module. It is responsible for distributing the intrusion detection problem among IDSs. Instead of having one IDS, there will be several IDSs each responsible for a portion of the Computational Grid. The cooperation module will be responsible for sharing the results obtained at each IDS among the other IDSs. This sharing will be achieved through a protocol that defines how will an IDS query and share its results with other IDSs.

This sharing can be either implemented using peer-to-peer techniques [3] or by using the Grid Information System (GIS) to share the results. Both techniques are distributed, do not rely on a central server and support the dynamic nature of Computational Grids.

The protocol used in this implementation is simple. Each IDS has a subset of users that are in its scope. For each user, the IDS will query other IDSs (peers) for their results of this user. Then the received results will be used to decide whether the user is intruder or normal. When an intruder is detected at an IDS, a warning will be sent to other IDSs to take appropriate actions.

4. Testing of GIDA

The performance of the proposed GIDA implementation is measured by five main parameters:

- **False positive percentage:** This measures the percentage of normal users that are miss classified by the system as intruders.
- **False negative percentage:** This measures the percentage of intruders that are miss classified by the system as normal users.
- **Training time:** The time needed to train the LVQ neural network.
- **Detection duration:** The time duration needed by the system to detect the intrusion.
- **Recognition percentage:** This measures the accuracy of the LVQ to correctly classify and recognize users with the absence of intruders.

The value of these parameters is affected by the environment in which the system is running. There are controllable issues such as data preprocessing and number of IDSs that must be adapted to best fit uncontrollable issues such as the number of users, number of resources, and number of intruders in a given environment. These issues are discussed below.

4.1. Data preprocessing

The records in the log file are preprocessed (Figure 4) before applying them to the LVQ by grouping several records that are in the same window. The window can take several forms (Figure 5) depending on whether it is controlled by a fixed number of records in each window (Type 1), a fixed time period for the window regardless of the number of records in this period (Type 2), or a hybrid window with both size and time limits that determines the number of records in the window depending on which limit is reached first (Type 3, 4, and 5). Generally, increasing the number of records in the window – by increasing window size or duration – decreases the false negative percentage; but meanwhile increases the false positive and the detection duration which is not desired. The hybrid window (Type 5) gave better results because it kept the number of records in the window at the desired value even when using multiple IDSs which resulted in fewer number of available records. The hybrid approach also kept the detection duration relatively constant. The window type slightly affected the training time. Results are shown in (Figure 6).

4.2. Number of IDSs

This is an important issue that shows the scalability of the system and that it is possible to distribute the intrusion detection problem among multiple IDSs. Increasing the number of IDSs increased the percentage of false positive (Figure 7.a). This is because fewer information is available to each IDS about the user behavior. Meanwhile it decreased the percentage of false negative (Figure 7.b) because among the few user actions monitored at an IDS detecting deviation form them is easier. This trade of between false positive and false negative percentages exist in all

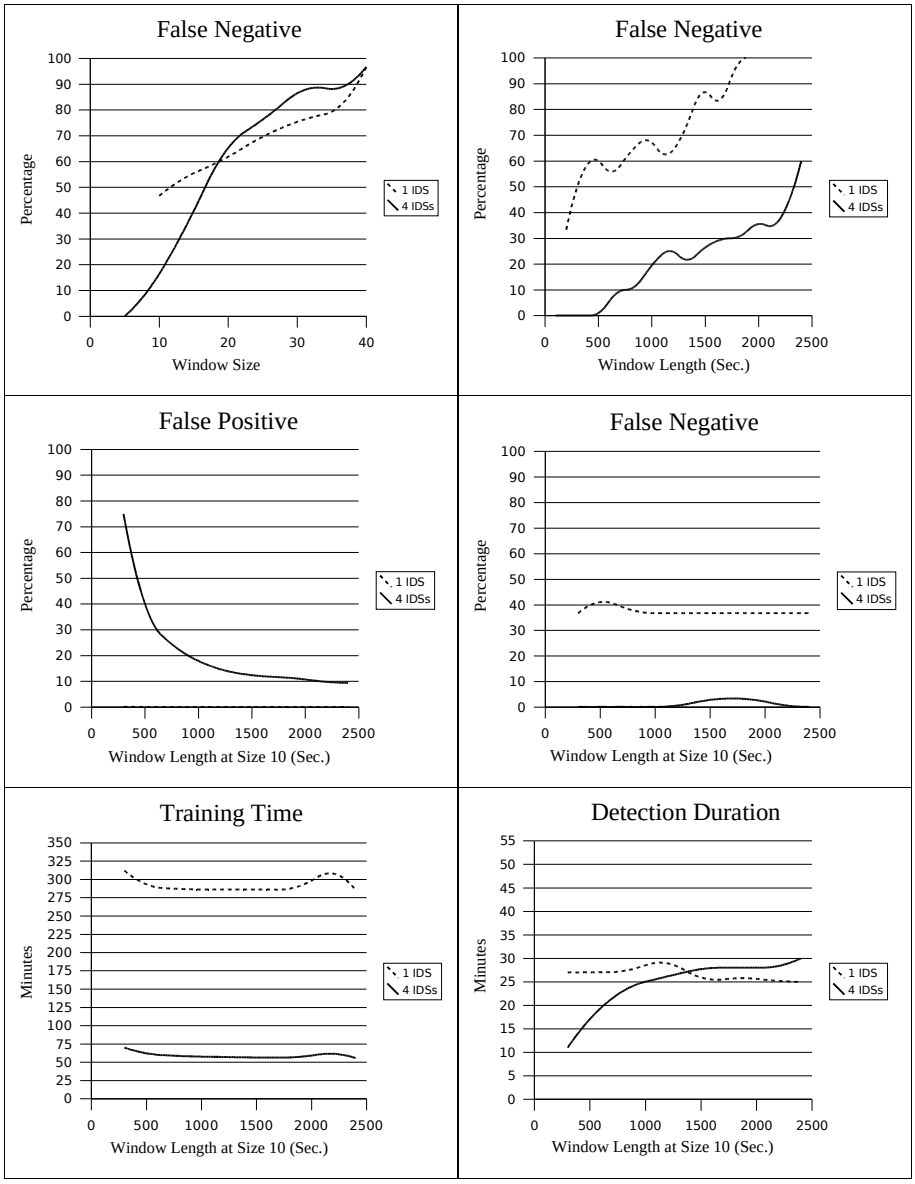

Figure 6. The Effect of window type.

intrusion detection systems. Increasing the number of IDSs has a great effect on reducing the training time (Figure 7.c) while only slightly decreasing the LVQ recognition (Figure 7.d). This shows that it is possible to distribute the intrusion detection problem but the number of IDSs must be carefully chosen to deliver the desired values of false positive and negative percentages. The detection duration was kept at an average of 25 minutes by using the hybrid window approach.

4.3. Number of users

This is another important issue that measure the scalability of the system at a specific configuration in accepting larger number of users. As shown in (Figure 8) increasing the number of users slightly increased the false positive percentage and reduced the false negative percentage. Centralized systems with one IDS was not scalable as training time increased exponentially, multiple IDSs kept training time low.

4.4. Number of resources

Increasing resources reduced the false positive percentage and increased the LVQ recognition while slightly affecting false negative percentage. This is because users have wider variety of resources to choose from and this gives them better distinct behavior. These results are shown in (Figure 9)

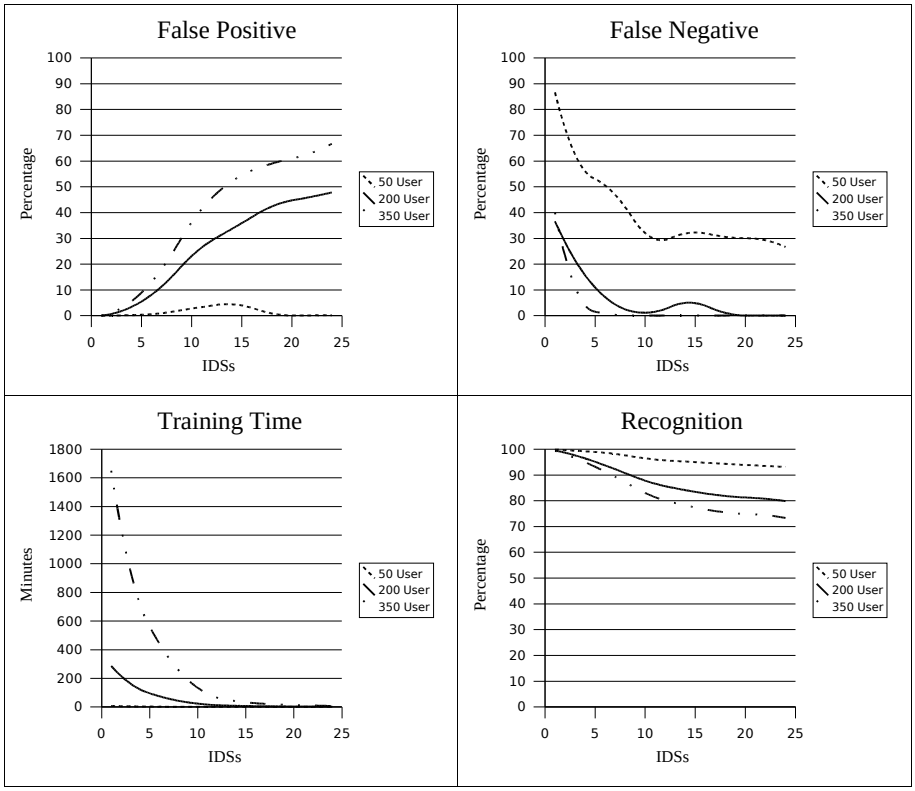

Figure 7. The Effect of the Number of IDSs.

4.5. Number of intruders

Increasing the number of intruders has only slightly increased the percentage of the false negative as shown in (Figure 10). It did not affect the other system parameters.

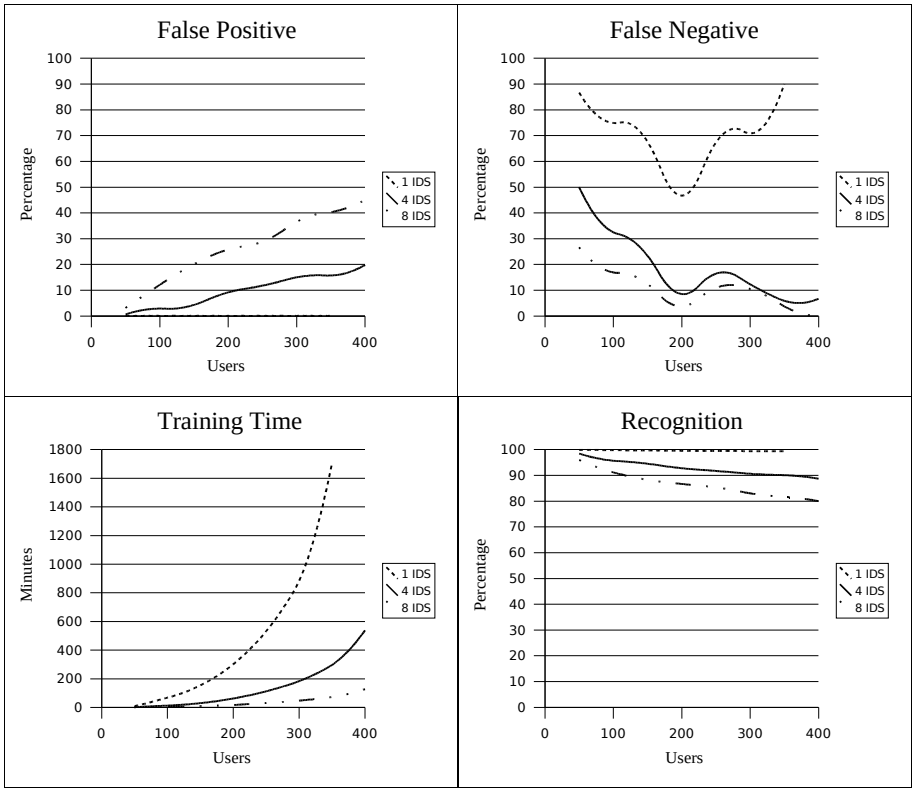

Figure 8. The Effect of Increasing the Users.

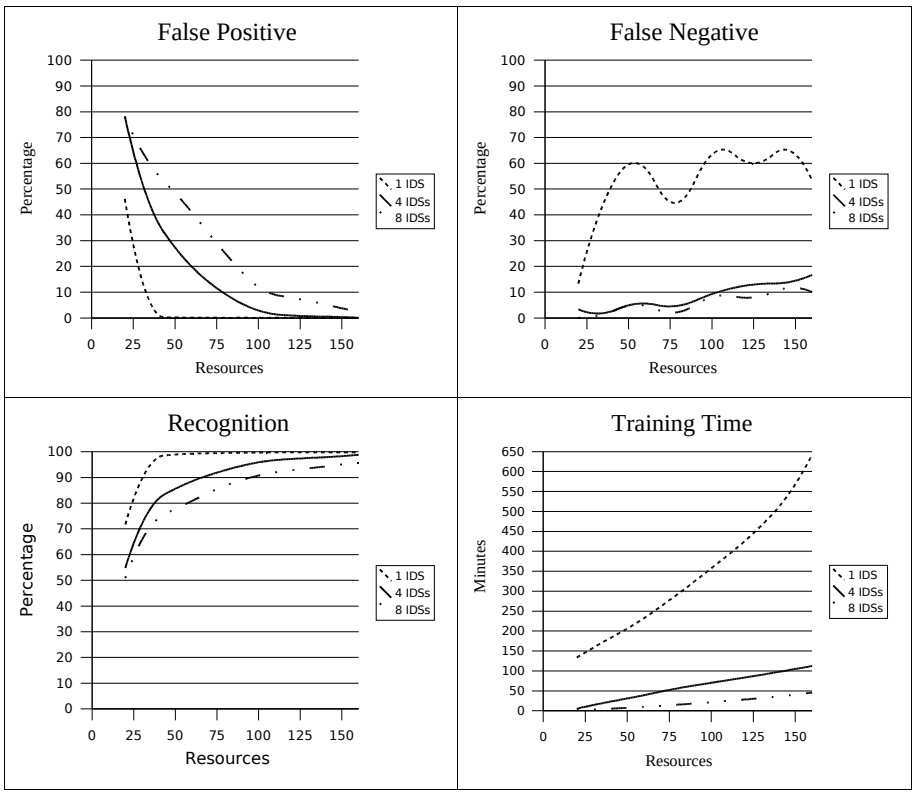

Figure 9. The Effect of the Number of Resources.

5. Conclusions and future work

Security is an important issue for the future of the Grid. As Grid technologies improve and real Grids start to appear, security will be more critical to protect the Grid resources in large collaborations and commercial applications. To increase security intrusion detection is needed as a second line of defense and to protect the Grid from insiders.

The proposed Grid Intrusion Detection Architecture (GIDA) is an open and flexible architecture that addresses the special requirements of the Grid. The implementation of this Architecture presented in this paper proved the applicability of such architecture in grid environments. The main issues affecting the system have been discussed to help in deciding the value of different parameters to increase the performance of the system in different Grid environments. This work helps to understand the problem of intrusion detection in Grid environments and to build future systems.

The effect of trust relationships between different resource owners and the use of heterogeneous IDSs should be further investigated. Also these two issues will raise a question about their effects on different QoSs and how these QoSs can be selected and measured. With Heterogeneous IDSs and trust relationships more complex algorithms will be needed for the cooperation module that will need further investigations. The application of the Grid in real problems

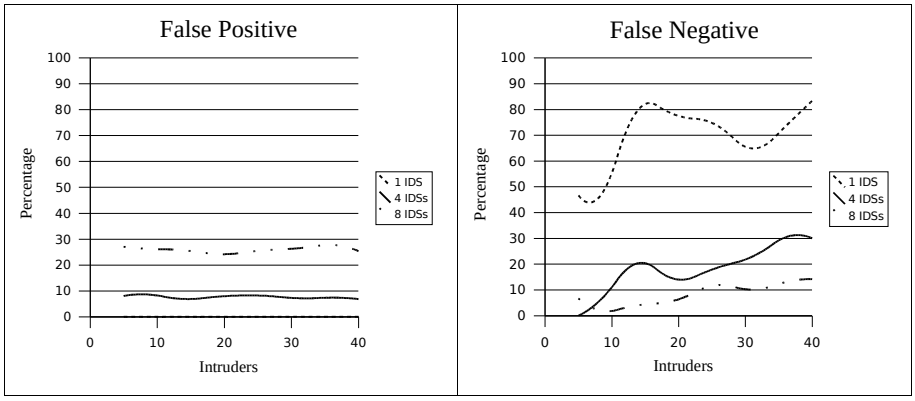

Figure 10. The Effect of the number of Intruders.

will help in building a knowledge base of attack signatures that will enable the use of misuse intrusion detection with the Grid.

6. References

[1] B. Segal, "Grid Computing: The European Data Grid Project", *IEEE Nuclear Science Symposium and Medical Imaging Conference*, Lyon, France, 15-20 October 2000.

[2] D. Brown, B. Suckow, and T. Wang, "A Survey of Intrusion Detection Systems", *CSE 221: Fall 2001 Projects, Department of Computer Science*, University of California, San Diego, USA, 2001.

[3] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu, "Peer-to-peer computing", *Technical Report HPL-2002-57*, HP Lab, 2002.

[4] E. Spafford, and D. Zamboni, "Data collection mechanisms for intrusion detection systems", *CERIAS Technical Report 2000-08*, CERIAS, Purdue University, 1315 Recitation Building, West Lafayette, IN, June 2000.

[5] Foster, I., and C. Kesselman (Eds), *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 1999.

[6] Grid Today. Daily News and Information for the Global Grid Community, July 22, 2002: Vol. 1 No. 6

[7] H. Casanova, "Simgrid: A Toolkit for the Simulation of Application Scheduling", *Proceedings of the First IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2001)*, Brisbane, Australia, May 15-18, 2001.

[8] H. Debar, "An Introduction to Intrusion-Detection Systems", *IBM Research, Zurich Research Laboratory*, Ruschlikon, Switzerland, 2000.

[9] H. Song, X. Liu, D. Jakobsen, R. Bhagwan, X. Zhang, K. Taura, and A. Chien, "The MicroGrid: a Scientific Tool for Modeling Computational Grids", *Proceedings of IEEE Supercomputing (SC 2000)*, Dallas, USA, Nov. 4-10, 2000.

[10] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, "A security architecture for computational grids", *Fifth ACM Conference on Computers and Communications Security*, November 1998.

[11] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid. Enabling Scalable Virtual Organizations", *International Journal of Supercomputer Applications*, 2001.

[12] J. Balasubramaniyan, J. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni, "An Architecture for Intrusion Detection using Autonomous Agents". *Department of Computer Sciences*, Purdue University, Coast TR 98-05; 1998.

[13] J. Marin, D. Ragsdale, and J. Surdu, "A Hybrid Approach to Profile Creation and Intrusion Detection", *in Proceedings of DARPA Information Survivability Conference and Exposition*, Anaheim, CA, 12-14 June 2001.

[14] M. Baker, R. Buyya, and D.Laforenza, "The Grid: International Efforts in Global Computing", *Intl. Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet (SSGRR'2000)*, Italy, 2000.

[15] M. Murshed, R. Buyya, and D. Abramson, "GridSim: A Grid Simulation Toolkit for Resource Management and Scheduling in Large-Scale Grid Computing Environments", *17th IEEE International Symposium on Parallel and Distributed Processing (IPDPS 2002)*, Fort Lauderdale, FL, USA, April 15-19, 2002.

[16] M. Tolba, I. Taha, and A. Al-Shishtawy, "An Intrusion Detection Architecture for Computational Grids", *First International Conference on Intelligent Computing and Information Systems*, Cairo, Egypt, June 2002.

[17] P. Anderson, "Computer Security Threat Monitoring and Surveillance", *Technical report*, James P. Anderson Company, Fort Washington, Pennsylvania, April 1980.

[18] R. Buyya, K. Branson, J. Giddy, and D. Abramson, "The Virtual Laboratory: Enabling On-Demand Drug Design with the World Wide Grid", *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid*, May 21-24, 2002.

[19] T. Kohonen, "Learning vector quantization", *MIT Press*, In M. Arbib, editor, The Handbook of Brain Theory and Neural Networks, 1995, pp. 537—540.

[20] The Global Grid Forum home page. http://www.gridforum.org/

[21] The Globus Project™ Homepage. http://www.globus.org/