AN INTRUSION DETECTION ARCHITECTURE FOR COMPUTATIONAL GRIDS

Prof. Dr. Mohammad F. Tolba Dr. Ismail A. Taha* Ahmad M. Al Shishtawy

Scientific Computing Department, Faculty of Computer and Information Sciences, Ain Shams University, Cairo.
tolba@asunet.shams.eun.eg

Abstract

Emerging scientific and business applications require access to large amounts of data and fast computational resources that may be available in distributed wide area networks. These resources are administrated locally and independently. This creates the need for computational grids, which manage this large collection of heterogeneous and dynamic resources in a scalable way. This paper reviews the unique security requirements of computational grid environments and shows the need for an intrusion detection system in grid environments. Moreover, differences between intrusion detection systems in grid environments and conventional distributed systems are illustrated. This paper introduces a proposed intrusion detection model for computational grid environments and presents how it can be used as part of one of the most leading grid research projects "the Globus Toolkit".

Keywords: *The Grid, Computational Grids, Security, Intrusion Detection, Globus Toolkit, GIDA.*

1. Introduction

In spite of the massive improvements in the computer hardware and networking, there are still problems that cannot be handled effectively with current generation of supercomputers [11]. That is due to their size, complexity, and/or variety of heterogeneous resources needed but are not available on a single machine. Therefore, the need for utilizing geographically distributed resources, as a single powerful computer becomes an emerging request. The challenge is to build a unified paradigm that simplify the development of distributed applications to link such resources and provide a reliable, scalable, flexible, and secured architecture.

The grid provides an easy way to couple resources that are geographically distributed, and belong to different administrative domains [12]. If a computer in Africa is connected to the grid, the user of this computer can for example run a program on a supercomputer in America, that uses an electronic telescope located in Australia, and store its results in a database system located in Europe. This allows a new generation of applications to be developed, but also creates a heaven for people that will try misuse this technologies to gain unauthorized access to these valuable resources, unless the designers of this emerging new technology bare in mind the strong security requirements of this technology.

Computational grids are also known by other names, such as, metacomputing, seamless scalable computing, global computing, and grid computing [3]. It is distinguished from conventional distributed computing by its focus on large-scale resource sharing, innovative applications, and in some cases, high-performance orientation [14]. The grid problem is defined in [14] as flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources. By the term resource we mean anything that can be allocated, including supercomputers, clusters, databases, storage systems, meteorological sensors and digital organizers.

There are three main issues that characterize computational grids: [3]

- **Heterogeneity:** The resources of the grid are heterogeneous. Having different architectures, operation systems, spanning multiple administrative domains that are geographically distributed.
- **Scalability:** The size of the grid can grow from just a few resources to span the entire Earth, so applications should deal with the degradation in performance as the grid increase in size.
- **Dynamicity or Adaptability:** In the grid with its huge number of resources, failure is the rule not the exception, so applications must deal with this dynamicity. Also different resources of the grid should be allowed to join or leave the grid whenever they want to.

Grid technologies complement rather than compete with existing technologies and it is considered to be the next generation Internet.

This paper first reviews security architecture in the grid environment and illustrates how it differs from traditional security systems. It introduces the Globus Toolkit, one of the most powerful research projects that seek to enable the construction of computational grids, and briefly discuss its main components. Then a discussion of the security status of the Globus Toolkit is presented and the need for intrusion detection is identified. Finally, this paper proposes an architecture that can be used to implement intrusion detection within the Globus Toolkit.

* Computer Department, Military Technical Collage, Cairo, Egypt.

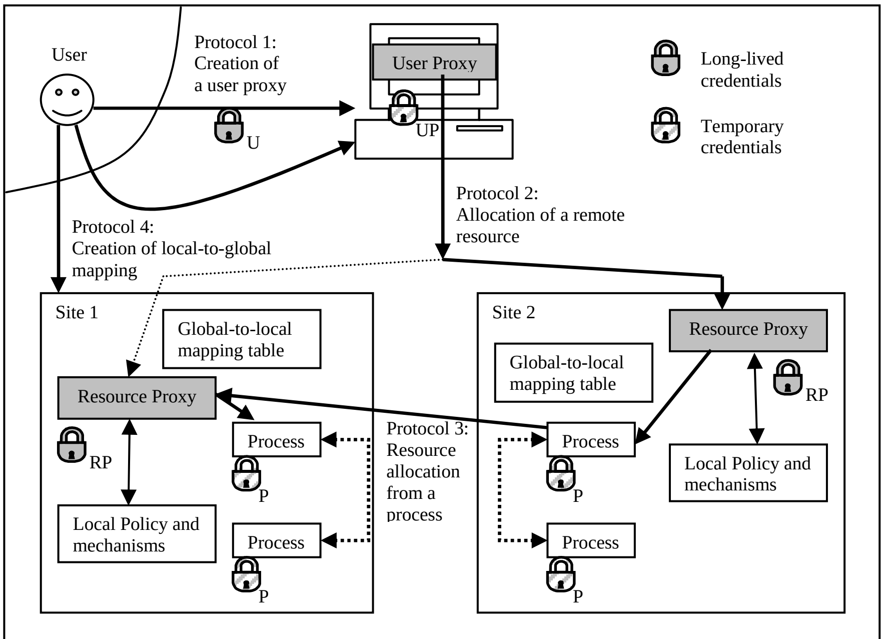

Figure 1: A typical computational grid security architecture.

2. Security in Grid Environments

Computational grids can be viewed as a large collection of resources, which are administrated locally and independently. Grid technologies couple these resources, and enable users to use them easily and efficiently in a way that was not possible before. Owners of these resources need to protect their valuable resources from misuse. So the grid must provide a secure place to pool resources without compromising the security at any site. Each group of these resources belongs to a single administration domain or a trusted domain. Each domain administrates and implements its own local security policy. The interdomain security of the grid should interoperate, rather than replace, or add any constraints, to these diverse local intradomain security policies.

Figure 1 depicts a general security architecture for a typical computational grid environment [13]. The curved line around the user indicates that he may leave after the user proxy is created. The main components of this architecture are described below:

- **User Proxy:** It is a session manager process given permission to act on behalf of a user for a limited period of time. This delegation is done by generating temporary credentials and giving it to the User Proxy. This provides the basic requirements for single sign on and the protection of user credentials.
- **Resource Proxy:** It is an agent used to translate between interdomain security operations and local intradomain mechanisms. Resource Proxy is part of Globus Resource Allocation Manager (GRAM) and it performs the mapping of authenticated Globus credentials into locally recognized credentials (Mapping from the global name to a local name).
- **Protocol 1:** Responsible for the creation of the user proxy and the generation of its temporary credentials that will allow it to act on behalf of the user.
- **Protocol 2:** Responsible for the remote allocation of resources. First a mutual authentication is performed between the user proxy and resource proxy, and then if the request is accepted a process is created and temporary credentials are created for this new process.
- **Protocol 3:** Responsible for resource allocation from a process. First the process authenticate itself to the user proxy and send its request, and then if the user proxy decide to accept the request, it uses protocol 2 to allocate that resource.
- **Protocol 4:** Each user has a global name and local name at each administrative domain. This protocol is responsible to map the global name to the local name.

The unique features of the grid added the following special constraints to normal security requirements [13]:

- Single sign-on.
- Protection of credentials.
- Interoperability with local security solutions.
- Exportability.
- Uniform credentials/certification infrastructure.
- Support for secure group communication.
- Support for multiple implementations.

Globus [21] is a research project that seeks to develop the fundamental technology needed to build computational grids. It is a collection of protocols, services, APIs and SDKs that are used together to build computational grids and support the development of its applications. The Globus Toolkit uses the *bag of services* approach, so it is a collection of services that developers of grid applications can choose from, according to their needs. Globus toolkit uses a layered architecture that takes the shape of an hourglass, at the neck is a set of well defined interfaces, which are used to access diverse local implementations of services and higher level services are build on top of those will defined interfaces. It is also an open source, open architecture system. The main components of the Globus Toolkit which are related to the work presented in this paper are [9, 10]:

- **Resource Management (GRAM):** Globus Resource Allocation Manager (GRAM) is at the bottom of the layered architecture [5]. It controls a set of resources at the same administrative domain, and is responsible for local resource management. It is often implemented using some local resource management system (Condor, LoadLeveler, fork daemon…). It provides a set of standard APIs that higher level tools use to allocate resources, and acts as an interface to these resources. GRAM is used as a building block for a global resource management architecture that includes resource brokers and co-allocators, and communicates using Resource Specification Language (RSL).
- **Information (MDS):** Information about the grid system structure and state is very important because of the dynamic nature of the grid. Toolkit components and applications must adapt their behavior in response to changes in the system. The Globus Metacomputing Directory Service (MDS) [6] supports rich information about different system components, which can be used to discover the current available components in the grid and their states.
- **Security (GSI):** The Grid Security Infrastructure (GSI) implements the above architecture. The GSI provides a single sign-on authentication service, with support for local control over access rights and mapping from global to local user identities [4].
- **Health and Status (HBM):** Heartbeat Monitor (HBM) provides a mechanism to monitor the health and status of a distributed set of processes [20]. It contains a client interface that allows a process to register with a HBM service, which in turn expects to receive regular heartbeats from the process and data-collector API which allows other process to obtain information about the status of registered processes.

3. Present Evaluation of Globus Toolkit Security Architecture

Today's computer systems are vulnerable both to abuse by insiders and to penetration by outsiders. Current security systems are not enough to protect systems from insiders and outsiders [19]. Passwords can be cracked, keys can be stolen, firewalls does not protect the system from insiders and outsiders can dig under the firewall, security systems contains bugs and holes that are impossible to fix in a feasible way were legitimate users can miss use their authority and privileges. Thus auditing is viewed as the last line of defense. Because of the above reasons and may be more the need for intrusion detection systems becomes a must. Intrusion detection systems are based on the assumption that the normal use of the system is different from malicious use [1]. They mainly analyze the auditing file and try to discover any suspicious user behavior, and take appropriate actions if such user is discovered.

The objective of the intruder is to gain access to a system (Authentication) or to increase the range of privileges accessible on a system (Authorization) [19]. This requires the intruder to acquire information that should have been protected. In the context of the Globus Toolkit – in which the security architecture (GSI) is currently based on Public Key Infrastructure – this information is in the form of user private key. In cases when some other user’s private key is known, an intruder can log in to a system and exercise all the privileges accorded to the legitimate user.

Users private keys can be protected in one of two ways [16]:

- Access Control: The private key is kept in a file accessible only by the private key owner account. For increased protection this file may also be encrypted using a password.
- Smart Cards: The private key may be stored on a smart card. This provides the best security but requires special hardware that is currently not widely used.

The Grid Security Infrastructure (GSI) assumes that no one - other than the owner of the private key - can gain access to the file containing the private key. Although the previous two ways are considered secure enough to protect the private key of the user, there still a small probability that flaws can occur and an intruder can gain access to this file. Techniques used to crack password protected systems such as trying default passwords or exhaustively trying all short passwords is not valid in GSI because with current technology it is not feasible to guess the user private key. The most important requirement in the GSI is the single sign on. This is important because the user application may require the use of hundreds of resources located at different domains, and requiring the user to manually sign on each of these domains is impractical. Another reason is that the user application may run for days or weeks, and we may not expect the user to sit in front of the computer and manually log on whenever a new resource is needed. The simplest way to solve this problem is that the user will give his application a copy of his private key. So the application will use this key to authenticate and gain authorized access to resources needed to complete its task. This simple method is never used for two reasons:

- The application can do whatever the user can do because it has a copy of the user's private key. And the user can not control what the application is allowed to do (for example in the case of the application failure it may enter an infinite loop requesting resources from different domains)
- More importantly, the program normally runs on remote computers, this means that a copy of the private key will exist at remote sites. This makes the job of an intruder very easy.

To solve this problem in the GSI uses delegation, the user may delegate some of his rights to his application. This delegation is done by producing temporary credentials (a certificate and a private key) signed by the user that the application will use to act on behalf of the user. The user can place any restrictions he wants on these temporary credentials, such as the period of time that they are valid, the sites that the application can communicate with and so on. Although if an intruder gain access to this temporary credential (temporary private key) is less dangerous than gaining access to the user private key, an intruder is still able to do harmful or unauthorized work using this temporary private key.

Currently, the Globus Toolkit and its Grid Security Infrastructure (GSI) do not have any intrusion detection system that can avoid any of the previously discussed security leaks. Hence, computational environments such as Globus Toolkit needs intrusion detection system to protect themselves from intruders that will try to misuse the valuable resources made available by the grid to any legitimate user from any location on earth.

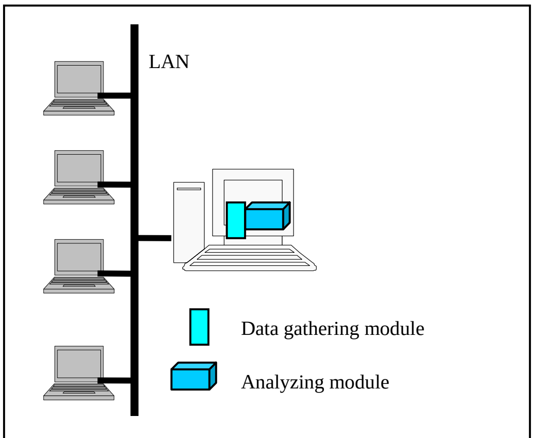

Figure 2: Centralized Intrusion Detection

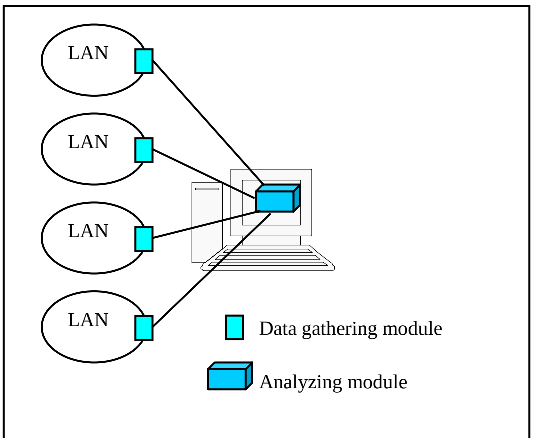

Figure 3: Distributed Intrusion Detection

4. Proposed Grid Intrusion Detection Architecture (GIDA)

Intrusion detection systems can be divided into two main modules. A data gathering module (DGM), that will gather needed information about the network and its users. And an analyzing module (AM), that will analyze gathered information in some way to detect intruders.

Traditionally the intrusion detection system with its two components (DGM and AM) is installed at the network server to monitor all passing data packets and determine suspicious connections (Figure 2). The problem with this architecture is that in the grid there is no single network server that can monitor all connections. Rather the grid consists of different domains, each is administrated locally, each running its own network server, and implementing its own security policy. So intrusion detection in the grid cannot be centralized at one server, but should be distributed in some way among different servers that coordinate and cooperate to detect intruders. This distributed architecture is also required to support the scalability and the dynamic nature of the grid.

Current distributed intrusion detection systems [15, 18] distributes the DGM among different local servers that gather information and send it to a centralized AM that is located at a secure place (Figure 3). This architecture is better than pure centralized architecture to deal with wide area networks, but still not suitable for the grid. Because the centralized AM creates a bottleneck and may become a hotspot and could provide a single point of failure that is not acceptable in the grid. Another problem with this distributed intrusion detection systems is that because of the large size of the grid and the complex trust relationships that exists between different administrative domains it is not possible to create a single central server that all domains will agree on and trust.

Moreover, the grid intrusion detection system should have a pre-knowledge about normal users behaviors as well as the different types of attacks and the corresponding set of suggested actions against each attack type. This problem arises due to the huge size of the grid - not simply a LAN or WAN - with millions of users with diverse behaviors and complex relationships. This makes the knowledge about normal user behavior very hard to obtain. Because grids are still under research, knowledge of different types of attacks that are unique to the grid environment is still not well defined. However, a grid intrusion detection system should monitor the user behavior outside its distributed administrative domains, and try to detect any suspicious behavior, and prevent the user from using the grid if he was identified to intrude any of provided services or resources at any point of time.

Based of the previous evaluation of the GSI, it can be concluded that intrusion detection in the grid cannot be centralized and both the DGM and AM must be distributed. Therefore, when designing an intrusion detection system for a grid architecture the following design issues should be satisfied:

- It must be scalable, to cope with the growing size of the grid.
- It must deal with the heterogeneous components of the grid.
- Its overhead should be minimal.
- It must support Integrity and confidentially because valuable auditing information will be transmitted across open networks not a local LAN.
- It must manage the different trusted relationships between different domains since the grid consists of different independent administrative domains.
- It must avoid bottlenecks and single point of failure.
- It must implement both the data gathering module (DGM) and the analysis module (AM) in a distributed fashion.

To implement such grid intrusion detection module in a grid architecture like the Globus Toolkit, it should provide a set of basic services on top of the Globus basic services that other higher level services use to achieve its goals.
These basic services and higher level services (Figure 4) include:

The Audit File

The audit file is one of the important elements of intrusion detection systems. Most intrusion detection techniques analyze information in this file to detect intruders. Because the lack of a central server, there are several problems that face the implementation of this file [17]. The following points provide the answers to these problems:

- **Where to keep this file?** The Globus toolkit provides an information service (MDS) that is a good place to store our auditing information.
- **How to gather this information?** A low-level service can be added to the Globus Toolkit. This service will be called the Intrusion Detection Agent (IDA) (Figure 4). The IDA will be installed at each administrative domain and will cooperate with GRAM and the Resource Proxy to gather information about the users and send it to the MDS. The IDA represents the DGM component in the proposed architecture. It is responsible to deal with the heterogeneity at each domain and provide a uniform view of the grid to higher services.
- **What information will be captured?** The audit file in this case will capture information like: the type of services requested, the location that the request came from, the time and frequencies of requests, whether the request came directly from the user or from the user proxy or from a process that is part of the user application, the type of resources allocated to the user, the type and frequency of errors and any security violations.
- **How will this information be moved to the MDS?** Some of the information may contain critical information that intruders should not know about. MDS provide an authenticated and encrypted way of communication that can be use to transfer this information. The Nexus communication library [7, 8] may be used, which provides multiple communication methods and security levels, with the capability of automatically choosing the best available method.
- **How will information stored in MDS be protected?** MDS provide mechanisms to restrict access to stored information. Access may be read only, write only or read and write. IDA will be allowed to write only in MDS.
- **Which MDS?** MDS is not centralized, it is distributed, and there exists many servers that contain information about the grid. Choosing MDS server will be based on trust relationship between the local administrative domain and the MDS server. This trust may be due to the choice of the local system administrator or by automatically trusting MDS servers whose certificates are signed by trusted certificate authorities. This information can be also sent to multiple MDS servers.
- **Push or Pull model?** Currently push model is used. That is IDA pushes its data to MDS at a uniform rate. A pull model may also be used, that is the AM will ask the DGM to send gathered information when it is needed.

The Intrusion Detection Server (IDS)

This is a high level service that uses the information gathered by IDA and stored in the MDS. The IDS represents the AM component in the proposed architecture. With each MDS server there will be an IDS attached to it. Before allowing IDA to store information inside MDS, it must first register with the attached IDS that will give IDA appropriate credentials so it can be identified. This registration process will define the scope of the IDS. The scope defines the part of the grid that this IDS sees and is able to analyze. IDS with larger scope will be able to analyze the behavior of users better than IDS with smaller scope. Moreover, the scope of IDSs may overlap. As said before in this large collection of resources the failure is the rule not the exception. The Heartbeat Monitor mentioned above can be used to check the statues of the distributed IDAs and IDSs periodically, using this information each IDA or IDS can adapt its behavior to deal with the failure of any of the other components.

Analyzing the audit data to detect intruders may be done in any of the intrusion detection approaches (Statistical Profile-based, Rule-based, Neural Networks, Expert Systems, State-Transition Analysis, Pattern Matching, Model Based, Network Security Monitor, Autonomous Agents or Data Mining Techniques) [2]. Each IDS may use different approach. When an intruder is discovered at any IDS, a warning will be sent to all registered IDAs, which will in turn warn the local security mechanism to take appropriate action(s). IDSs may cooperate with each other, sending warning to other IDSs that share a common user if this user is detected as an intruder at one IDS. They may also share information about common users.

Cooperation with Local Intrusion Detection Systems

If the local administrative domain has an intrusion detection mechanism, it can cooperate with GIDA. If an intruder is detected locally, the local intrusion detection system will warn the IDA which will send this warning to the registered IDSs and they will forward this warning to all other IDAs and IDSs that share this user. Then any IDA that receives this warning will automatically warn the local security system to take the desired action(s).

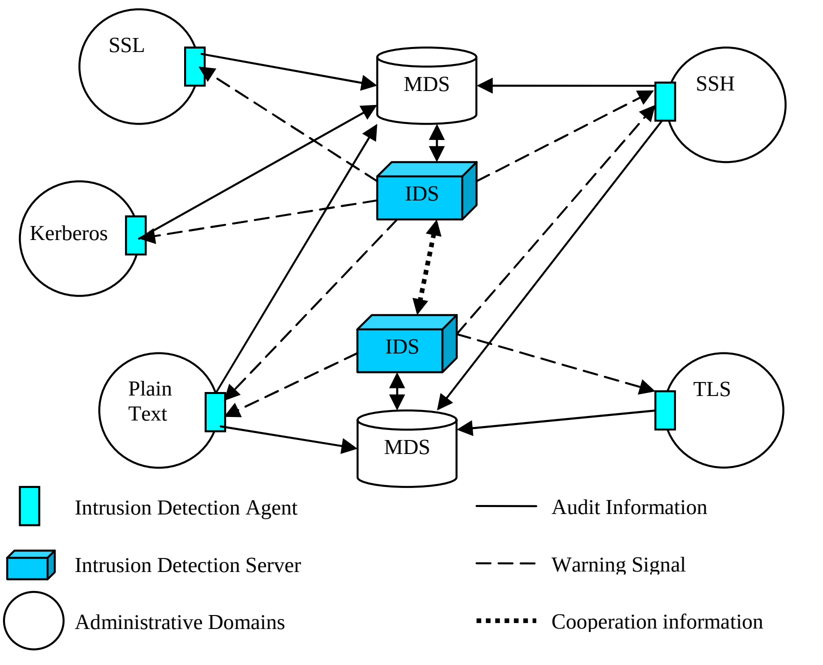

Figure 4: Grid Intrusion Detection Architecture

5. Conclusion

No security system exists that is free from bugs and holes, and it is not feasible with current technology to create such system. So intrusion detection is considered a vital part of any security system, especially grid computational environments, as a second line of defense to detect intruders who managed to penetrate security systems, and authorized users that misuse their authority. The Grid Security Infrastructure GSI used by the Globus Toolkit needs intrusion detection. Because of the special characteristics of the grid, traditional intrusion detection mechanisms cannot be used without modifications and it must be a coordinated and cooperative task of distributed grid components/resources. The proposed Grid Intrusion Detection Architecture (GIDA) solves the problems found in previous centralized and distributed intrusion detection systems and can play a very important role in any grid security system. The proposed GIDA may help in resolving several problems of distributed intrusion detection systems if they are implemented in a grid environment. This paper presented the proposed GIDA in the context of the Globus Toolkit but it can be implemented in any other grid system. The proposed GIDA architecture provides a good start point for discussion and further investigations.

References

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

[2] Badr S., ***Security Architecture for Internet Protocols.*** Ph.D. dissertation, Military Technical Collage, Cairo, Egypt, 2002.

[3] Baker M., Buyya R., Laforenza D., ***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.

[4] Butler R., Engert D., Foster I., Kesselman C., Tuecke S., Volmer J., and Welch V., ***Design and deployment of a national-scale authentication infrastructure.*** Submitted, 1999.

[5] Czajkowski K., Foster I., Kesselman C., Martin S., Smith W., and Tuecke S., ***A resource management architecture for metacomputing systems.*** Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1997.

[6] Fitzgerald S., Foster I., Kesselman C., Laszewski G., Smith W., and Tuecke S., ***A Directory Service for Configuring High-Performance Distributed Computations.*** In Sixth IEEE International Symposium on High Performance Distributed Computing, 1997.

[7] Foster I., Geisler J., Kesselman C., and Tuecke S., ***Managing multiple communication methods in high-performance networked computing systems.*** Journal of Parallel and Distributed Computing, 1997.

[8] Foster I., Karonis N. T., Kesselman C., Tuecke S., ***Managing Security in High-Performance Distributed Computing.*** Cluster Computing, 1998.

[9] Foster I. and Kesselman C., ***Globus: A Metacomputing Infrastructure Toolkit'.*** Proceedings of the Workshop on Environments and Tools for Parallel Scientific Computing, SIAM, Lyon, France, August 1996.

[10] Foster I. and Kesselman C., ***The Globus Project: A Status Report.*** Proceedings of the Seventh Heterogeneous Computing Workshop, March 1998, IEEE Computer Society Press.

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

[12] Foster, I., Kesselman, C., Nick, J. and Tuecke, S., ***The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.*** Globus Project, 2002.

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

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

[15] Heberlein L.T., Levitt K. and Mukherjee B., ***Internetwork Security Monitor: An Intrusion-Detection System for Large-Scale Networks.*** In Proc. 15th National Computer Security Conference, Oct. 1992.

[16] Novotny J., Tuecke S., Welch V., ***An Online Credential Repository for the Grid: MyProxy.*** 10th IEEE Symp. On High Performance Distributed Computing, 2001.

[17] Porras P. ***STAT: A State Transition Analysis Tool for Intrusion Detection.*** Master's thesis, Computer Science Department, University of California, Santa Barbara, June 1992.

[18] Snapp S., Brentano J., et al, ***A system for distributed intrusion detection.*** IEEE COMPCON, 1991.

[19] Stallings W., ***Network and Internetwork Security - Principles and Practice***, Prentice Hall, 1995.

[20] Stelling P., Foster I., Kesselman C., Lee C., and Laszewski G., ***A Fault Detection Service for Wide Area Distributed Computations.*** Proc. 7th IEEE Symp. on High Performance Distributed Computing, 1998.

[21] www.globus.org.