ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Grid Computing: A solution to processing time and power issues

Ms. B. Sornalatha

IV BE (CSE), RVS College of engineering and technology, Dindigul-5, Tamilnadu, India[1]

ABSTRACT— Grid computing is a method of harnessing the power of many computers in a network to solve problems requiring a large number of processing cycles and involving huge amounts of data. The grid computing helps in exploiting underutilized resources, achieving parallel CPU capacity; provide virtual resources for collaboration and reliability. But, large-scale deployment of grids will occur when users can count on their security.

On the grid, participation often takes place at the organizational rather than the individual level. Thus, expressing restrictive policies on a user-by-user basis often proves difficult. Grid security is a multidimensional problem. Organizations participating in grids must use appropriate policies to harden their Infrastructures while enabling interaction with outside resources.

This paper will describe the reasons for using grid and analyze the security requirements of large-scale grid computing. We propose a security policy for grid systems that addresses requirements for single sign-on, interoperability with local-policies, and describe security architecture and associated protocols.

KEYWORDS— Heterogeneity, Resource allocation, Virtual computing, Authentication.

I. INTRODUCTION

Grid applications are distinguished from traditional client server applications by their simultaneous use of large numbers of resources, dynamic resource requirements, use of resources from multiple administrative domains, complex communication structures and stringent performance requirements, among others. Although commercial and research organizations might have collaborative or monetary reasons to share resources, they are unlikely to adopt such a distributed infrastructure until they can rely on the confidentiality of the communication, the integrity of their data and resources, and the privacy of the user information.

This policy focuses on authentication of users, resources, and processes and supports user-to resource, resource to user, process-to-resource, and Process to process authentication. Also, frequently a single transaction takes place across many grid nodes that are dynamic and unpredictable. Finally, unlike the Internet, a grid gives outsiders complete access to a resource, thus increasing the security risk.

Most organizations today deploy firewalls around their computer networks to protect their sensitive proprietary data.

While scalability, performance and heterogeneity are desirable goals for any distributed system, the characteristics of computational grids lead to security problems that are not addressed by existing security technologies for distributed systems. For example parallel computations that acquire multiple computational resources introduce the need to establish security relationships not simply between a client and a server, but among potentially hundreds of processes that collectively span many administrative domains. Furthermore, the dynamic nature of grid can make it impossible to establish trust relationships between sites prior to application execution. Finally, by inter domain security solutions used for grids must be able to inter operate with, rather than replace, the diverse intra domain access control technologies inevitable encountered in individual domains.

In this paper, we describe new techniques that overcome many of the cited difficulties. We propose a security policy for grid systems that addresses requirements for single sign-on, inter operability with local policies, and dynamically varying

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

resource requirements. This policy focuses on authentication of users, resources, and processes and supports user-to-resource, resource-to-user, process-to-resource, and process-to-process authentication.

II. REASONS FOR USING GRID COMPUTING

When you deploy a grid, it will be to meet a set of customer requirements. To better match grid computing capabilities to those requirements, it is useful to keep in mind the reasons for using grid computing.

A. EXPLOITING UNDERUTILIZED RESOURCES

The easiest use of grid computing is to run an existing application on a different machine. The machine on which the application is normally run might be unusually busy due to an unusual peak in activity. The job in question could be run on an idle machine elsewhere on the grid. There are at least two prerequisites for this scenario. First, the application must be executable remotely and without undue overhead. Second, the remote machine must meet any special hardware, software, or resource requirements imposed by the application.

For example, a batch job that spends a significant amount of time processing a set of input data to produce an output set is perhaps the most ideal and simple use for a grid. If the quantities of input and output are large, more thought and planning might be required to efficiently use the grid for such a job. It would usually not make sense to use a word processor remotely on a grid because there would probably be greater delays and more potential points of failure.

In most organizations, there are large amounts of underutilized computing resources. Most desktop machines are busy less than 5 percent of the time. In some organizations, even the server machines can often be relatively idle. Grid computing provides a framework for exploiting these underutilized resources and thus has the possibility of substantially increasing the efficiency of resource usage.

The processing resources are not the only ones that may be underutilized. Often, machines may have enormous unused disk drive capacity. Grid Computing, more specifically, a "data grid", can be used to aggregate this unused storage into a much larger virtual data store, possibly configured to achieve improved performance and reliability over that of any single machine.

If a batch job needs to read a large amount of data, this data could be automatically replicated at various strategic points in the grid. Thus, if the job must be executed on a remote machine in the grid, the data is already there and does not need to be moved to that remote point. This offers clear performance benefits. Also, such copies of data can be used as backups when the primary copies are damaged or unavailable.

B. PARALLEL CPU CAPACITY

The potential for massive parallel CPU capacity is one of the most attractive features of a grid. In addition to pure scientific needs, such computing power is driving a new evolution in industries such as the bio-medical field, financial modeling, oil exploration, motion picture animation, and many others.

The common attribute among such uses is that the applications have been written to use algorithms that can be partitioned into independently running parts. A CPU intensive grid application can be thought of as many smaller "sub jobs," each executing on a different machine in the grid. To the extent that these sub jobs do not need to communicate with each other, the more "scalable" the application becomes. A perfectly scalable application will, for example, finish 10 times faster if it uses 10 times the number of processors.

Barriers often exist to perfect scalability. The first barrier depends on the algorithms used for splitting the application among many CPUs. If the algorithm can only be split into a limited number of independently running parts, then that forms a scalability barrier. The second barrier appears if the parts are not completely independent; this can cause contention, which can limit scalability.

For example, if all of the sub jobs need to read and write from one common file or database, the access limits of that file or database will become the limiting factor in the application's scalability. Other sources of inter-job contention in a parallel grid application include message communications latencies among the jobs, network communication capacities, synchronization protocols, input-output bandwidth to devices and storage devices, and latencies interfering with real-time requirements. Other sources of inter-job content in parallel grid application include message communications latencies among the jobs.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

C. VIRTUAL RESOURCES AND VIRTUAL ORGANIZATIONS FOR COLLABORATIONS

Another important grid computing contribution is to enable and simplify collaboration among a wider audience. In the past, distributed computing promised this collaboration and achieved it to some extent. Grid computing takes these capabilities to an even wider audience, while offering important standards that enable very heterogeneous systems to work together to form the image of a large virtual computing system offering a variety of virtual resources. The users of the grid can be organized dynamically into a number of virtual organizations, each with different policy requirements. These virtual organizations can share their resources collectively as a larger grid.

Sharing starts with data in the form of files or databases. A "data grid" can expand data capabilities in several ways. First, files or databases can seamlessly span many systems and thus have larger capacities than on any single system. Such spanning can improve data transfer rates through the use of striping techniques. Data can be duplicated throughout the grid to serve as a backup and can be hosted on or near the machines most likely to need the data, in conjunction with advanced scheduling techniques.

Sharing is not limited to files, but also includes many other resources, such as equipment, software, services, licenses, and others. These resources are "virtualized" to give them a more uniform interoperability among heterogeneous grid participants.

D. RELIABILITY

High-end conventional computing systems use expensive hardware to increase reliability. They are built using chips with redundant circuits that vote on results, and contain much logic to achieve graceful recovery from an assortment of hardware failures. The machines also use duplicate processors with hot plug ability so that when they fail, one can be replaced without turning the other off. Power supplies and cooling systems are duplicated. The systems are operated on special power sources that can start generators if utility power is interrupted. All of this builds a reliable system, but at a great cost, due to the duplication of high-reliability components.

In the future, we will see a complementary approach to reliability that relies on software and hardware. A grid is just the beginning of such technology. The systems in a grid can be relatively inexpensive and geographically dispersed. Thus, if there is a power or other kind of failure at one location, the other parts of the grid are not likely to be affected. Grid management software can automatically resubmit jobs to other machines on the grid when a failure is detected. In critical, real-time situations, multiple copies of the important jobs can be run on different machines throughout the grid. Their results can be checked for any kind of inconsistency, such as computer failures, data corruption, or tampering. Such grid systems will utilize "autonomic computing." This is a type of software that automatically heals problems in the grid, perhaps even before an operator or manager is aware of them. In principle, most of the reliability attributes achieved using hardware in today's high availability systems can be achieved using software in a grid setting in the future.

E. RESOURCE BALANCING

A grid federates a large number of resources contributed by individual machines into a greater total virtual resource. For applications that are grid-enabled, the grid can offer a resource balancing effect by scheduling grid jobs on machines with low utilization. This feature can prove invaluable for handling occasional peak loads of activity in parts of a larger organization. This can happen in two ways: An unexpected peak can be routed to relatively idle machines in the grid and if the grid is already fully utilized, the lowest priority work being performed on the grid can be temporarily suspended or even cancelled and performed again later to make room for the higher priority work.

Without a grid infrastructure, such balancing decisions are difficult to prioritize and execute. Occasionally, a project may suddenly rise in importance with a specific deadline. A grid cannot perform a miracle and achieve a deadline when it is already too close. However, if the size of the job is known, if it is a kind of job that can be sufficiently split into sub jobs, and if enough resources are available after preempting lower priority work, a grid can bring a very large amount of processing power to solve the problem. In such situations, a grid can, with some planning, succeed in meeting a surprise deadline.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

III. SECURITY IN GRID COMPUTING

A. The Grid Security Problem

We introduce of grid security problem with and example illustrated in figure1. We imagine a scientist, a member of a multi-institutional scientific collaboration, who receives e-mail from a colleague regarding a new data set. He starts an analysis program, which dispatches code to the remote location where the data is stored (site C). Once started, the analysis program determines that it needs to run a simulation in order to compare the experimental results with predictions. Hence, it contacts a resource broker service maintained by the collaboration (at site D), in order to locate the resources that can be used for the simulation. The resource broker in turn initiates
Computation on computers at two sites (E and G). These computers access parameter values store on a File system at another site (F) and also communicate among themselves and with broker, the original site, And the user.

We imagine a scientist, a member of a multi-institutional scientific collaboration, who receives e-mail from a colleague regarding a new data set. He starts an analysis program, which dispatches code to the remote location where the data is stored (site C). Once started, the analysis program determines that it needs to run a simulation in order to compare the experimental results with predictions. Hence, it contacts a resource broker service maintained by the collaboration (at site D), in order to locate idle resources that can be used for the simulation. The resource broker in turn initiates computation on computers at two sites (E and G). These computers access parameter values stored on a file system at yet another site (F) and also communicate among themselves (perhaps using specified protocols, such as multicast) and with the broker, the original site, and the user.

This example illustrates many of the distinctive characteristics of the grid computing environment:
The user population is large and dynamic. Participants in such virtual organizations as this scientific collaboration will include members of many institutions and will change frequently.
The resource pool is large and dynamic. Because, individual institutions and users decide whether and when to contribute resources, the quantity and location of available resources can change rapidly.
A computation may require, start processes on, and release resources dynamically during its execution. Even in our simple example, the computation required resources at five sites. In other words, throughout its lifetime, a computation is composed of a dynamic group of processes running on different resources and sites.
The processes constituting a computation may communicate by using a variety of mechanisms, including unicast and multicast. While these processes form a single logical entity, low-level communication connection may be created and destroyed dynamically during program execution.
Resources may require different authentication and authorization mechanisms and policies, which we will have limited ability to change. In figure 1, we indicate this situation by showing the local access control policies that apply at the different sites.
An individual user will be associated with different local name spaces, credentials, or accounts, at different sites, for the purposes of accounting and access control.

Resources and users may be located in different countries.

To summarize, the problem we face is providing security solutions that can allow computations, such as the one just described, to coordinate diverse access control policies and to operate securely in heterogeneous environments.

Single sign-on: A user should be able to authenticate once (e.g., when starting a computation) and initiate computations that acquire resources, use resources, release resources, and communicate internally, without further authentication of the user.

Protection of credentials: User credentials (passwords, private keys, etc.) must be protected.

Interoperability with local security solutions: While our security solutions may provide inter domain access mechanisms, access to local resources} will typically be determined by a loc~ security policy that is enforced by a local security mechanism. It is impractical to modify every local resource to accommodate inter domain access; instead, one or

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

more entities in a domain (e.g., inter domain security servers) must act as agents of remote clients/users for local resources.

Exportability: We require that the code be (a) exportable and (b) executable in multinational test beds. In short, the exportability issues mean that our security policy cannot directly or indirectly require the use of bulk encryption.

Uniform credentials/certification infrastructure: Inter domain access requires, at a minimum, a common way of expressing the identity of a *security principal* such as an actual user or a resource. Hence, it is imperative to employ a standard (such as X.509v3) for encoding credentials for security principals.

Support for secure group communication.

A computation can comprise a number of processes that will need to coordinate their activities as a group.

Support for multiple implementations:

The security policy should not dictate a specific implementation technology. Further, it should be possible to implement the security policy with a range of security technologies, based on both public and shared key cryptography.

1. ***Authentications:*** the process by which a subject proves its identity to a requester, typically through the use of a credential. Authentication in which both parties (i.e., the requester and the requester) authenticate themselves to one another simultaneously is referred to as *mutual authentication.*

2. An ***object*** *is a* resource that is being protected by the security policy.

3. A ***trust domain*** *is a* logical, administrative structure within which a single, consistent local security policy holds.

With these terms in mind, we define our security policy as follows:

1) The grid environment consists of multiple trust domains.
2) Operations that are confined to single trust domain are subject to local security policy only.
3) Both global and local subjects exist. For each trust domain, there exists a partial mapping from global to local subjects.
4) Operations between entities located in different trust domains require mutual authentication.
5) An authentication global subject mapped into a local subject is assumed to be equivalent to being locally authenticated as the local subject.
6) All access control decisions are made locally on the basis of the local subject.

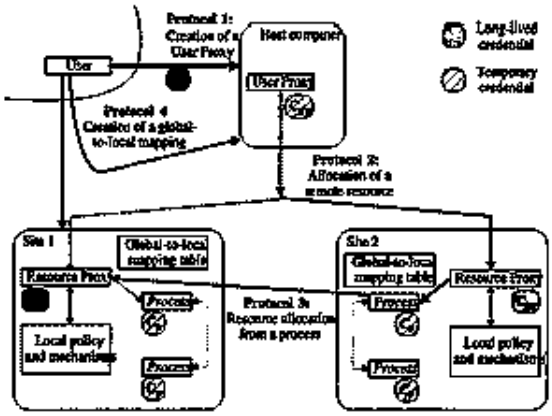

Figure 2: A computational grid security architecture.

7) A program or process is allowed to act on behalf of a user and be delegated a subset of the user's rights.
8) Processes running on behalf of the same subject with in the same trust domain may share a single set of credentials.

IV. GRID SECURITY ARCHITECTURE

The security policy defined in section c provides a context with in which we can construct specific security architecture. In doing so, we specify the set of subjects and objects that will be under the jurisdiction of the security policy and define the protocols that will govern interactions between these subjects and objects. Fig 2 shows an overview of out security architecture.

We are interested in computational environments so the objects in our architecture would be computers, data repositories, networks and so forth. The subjects are users and processes.

Grid computations may grow and shrink dynamically acquiring resources when required to solve a problem and releasing them when they are no longer needed. Each time a computation obtains a resource, it does so on behalf of a particular user. However, it is frequently impractical for that "user" to interact directly with each such resource for the purposes of authentication: the number of resources involved may be large, or, because some applications may run for extended period

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

of time, the user may wish to allow a computation to operate without intervention. Hence, we introduce the concept of a *user proxy* that can act on a user's behalf without requiring user intervention.

Definition: A user proxy is a session manager process given permission to act on behalf of a user for limited period of time.

Within the architecture, we also define an entity that represents a resource, serving as the interface between the grid security architecture and the local security architecture.

Definition: A resource proxy is an agent used to translate between interdomain security operations and local intradomain mechanisms.

Notations used in the rest of the paper are shown in within our architecture; we meet the above requirement by allowing a user to "logon" to the grid system, creating a user proxy using Protocol

1. The user proxy can then allocate resources using protocol 2. Using protocol 3. a process created can allocate additional resources directly. Finally, Protocol 4 can be used to define a mapping from a global to local subject.

V. CONCLUSION

Traditional computing environments don't provide flexibility for sharing resources to form "virtual organizations". Grid computing provides a promising and efficient way of using computing and storage resources. It serves as "Computing on Demand" model similar to the way electrical power is used. Ideal for collaborative environments because it provides dynamic resource sharing among different geographic locations and also it hides the complexity from the user who will see the grid as a huge computing and storage device.

We have described security architecture for large-scale distributed computations. The architecture is immediately useful and, in addition, provides a firm foundation for investigations of more sophisticated mechanisms.

Our architecture addresses most of the requirements introduced. The introduction of a user proxy addresses the single sign-on requirement and also avoids the need to communicate user credentials. The resource proxy enables interoperated with local security solutions, as the resource proxy can translate between inter-domain and intra-domain security solutions. Because encryption is not used within the associated protocol, export control issues and hence international use are simplified.

The security design presented addresses a number of scalability issues. The sharing of credentials by processes created by a single resource allocation request means that the establishment of process credentials will not, we expect, be a bottleneck. The fact that resource allocation requests must pass via the user proxy is a potential bottleneck this must be evaluated in realistic applications and, if required, addressed in future work. One major scalability issue that is not addressed is the number of users and resources. Clearly, other approaches to the establishment of global to local mappings ~ be required when the number of users and/or resources are large on example is the use-condition approaches to authorization. However, we believe the current approach can deal with this.

REFERENCES

- G Foster and C. Kasehnan, editors. *Computational Grids: The Future of High Performance Distributed Computing.* Morgan Kaubann, 1998.
- I. Foster and C. Kessehnan. The Globus project: A progress report. In *Heterogeneous Computing Workshop,* March 1998.
- I. Foster, C. Kessehnan, and S. Tuecke. The Nexus approach to integrating, multithreading and communication. *Journal of Parallel and Distributed Computing.*