A Security Policy in GRID Architecture

Alexander Kemalov

Abstract: ***A major requirement for grid computing is security. At the base of any grid environment, there must be mechanisms to provide security including authentication, authorization, data encryption and so on. Emerging distributed applications require fast access to large quantities of data and extremely fast computational resources. Both resources and data are often distributed in a wide-area network-Internet with components administered locally and independently.***

Key words: ***Computer Systems, Client-server, Security, Grid, Authentication, Authorization***

INTRODUCTION

Grid computing is gaining a lot of attention within the IT industry. Though it has been used within the academic and scientific community for some time, standards, enabling technologies, toolkits and products are becoming available that allow businesses to utilize and reap the advantages of grid computing.

Grid applications are distinguished from traditional client-server applications by their simultaneous use of large numbers of resources, dynamic resource requirements, and use of resources from multiple administrative domains, complex communication structures, among others.

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

In this paper, we propose a security policy for grid systems that addresses requirements for single sign-on, interoperability with local policies, and dynamically varying resource requirements. This policy focuses on authentication of users, resources and processes. We also describe security architecture.

SECURITY PROBLEMS IN GRID ARCHITECTURE

Let us take a typical scenario of large-scale distributed communications in Grid architecture – fig.1. User A as a member of multi-places public cooperation, who receives with e-mail information regarding a data set. He starts an analysis program, which dispatches code to the remote location where the data is stored (Place 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 Place D), in order to locate idle resources that can be use for the simulation. The resource broker in turn initiates computation on computers at two places (E and G).

These computers access parameter values stored on a file system at yet another place (F) and also communicate between themselves, the original place, and the user. This example illustrates many of the distinctive characteristics of the grid computing environment:

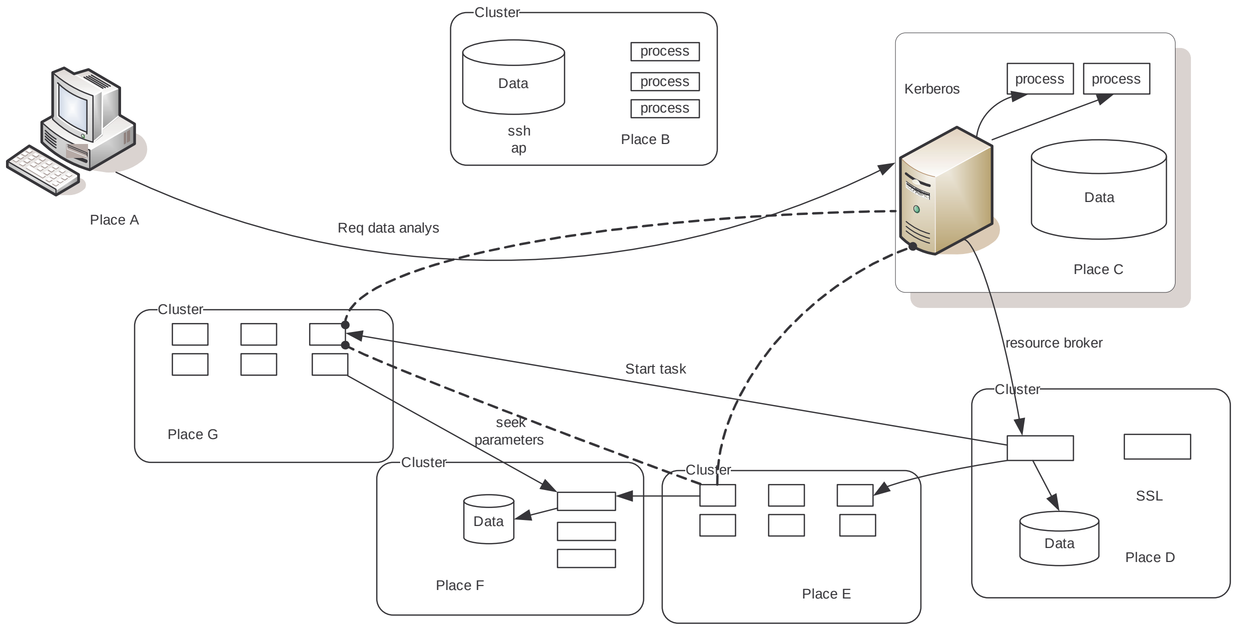

fig.1

• The user population is large and dynamic. Participants in such virtual organizations as this distributed collaboration will include members of many institutions and will change frequently.

• The resource pool is large and dynamic. Because individual places and users decide when to contribute resources, the quantity and location of available resources can change rapidly.

• A computation or processes may acquire, start processes on, and release resources dynamically during its execution. Even in our simple example, the computation acquired resources at five places. In other words, throughout its lifetime, a computation is composed of a dynamic group of processes running on different resources and places.

• The processes start a computation may communicate by using a variety of mechanisms, including unicast and multicast. While these processes form a single, fully connected logical entity, low-level communication connections (e.g., TCP/IP sockets) may be create and destroy dynamically during program execution.

• Resources may require different authentication and authorization mechanisms and policies, which we will have limited ability to change. In fig. 1, we indicate this situation by showing the local access control policies that apply at the different places. These include Kerberos, passwords, Secure Socket Library (SSL).

• An individual user will be associated with different local name spaces, credentials, or accounts, at different places, for the purposes of accounting and access control. At some sites, a user may have a regular account. The user may use a dynamically assigned guest account or simply an account created for the collaboration.

To summarize, the problem 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.

SECURITY REQUIREMENTS IN GRID

Grid systems and applications may require any of the standard security functions, including authentication, access control, integrity, privacy. In this paper, we focus primarily on issues of authentication and access control. Specifically, we seek to provide authentication solutions that allow a user, the processes that comprise a user's computation, and the resources used by those processes, to verify each other's identity. In the second place - allow local access control mechanisms to be applied without change, whenever possible. Authentication forms the foundation of a security policy that enables diverse local security policies to be integrated into a global framework.

In developing a security architecture that meets these requirements, we also choose to satisfy the following constraints derived from the characteristics of the grid environment and grid applications:

Single sign-on: A user should be able to authenticate 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 interdomain access mechanisms, access to local resources will typically be determined by a local security policy that is enforced by a local security mechanism. One or more entities in a domain (e.g., interdomain security servers) must act as agents of remote clients/users for local resources.

Exportability: We require that the code be exportable and executable in multinational test beds. The exportability issues mean that a security policy cannot directly or indirectly require the use of bulk encryption.

Uniform credentials/certification infrastructure: Interdomain access requires 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 (X.509) 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. The composition of a process group will change during the lifetime of a computation and is needed for secure communication for dynamic groups.

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

A GRID SECURITY POLICY

A security policy is a set of rules that define the security subjects, objects and relationships among them. While many different security policies are possible, we present a specific policy that addresses the issues introduced in the above section while reflecting the needs and expectations of applications, users, and resource owners. The following points represent the first such grid security policy that has been to define to this level:

- ***A subject*** is a participant in a security operation. In grid systems, a subject is generally a user, a process operating on behalf of a user, a resource (such as a computer or a file), or a process acting on behalf of a resource.
- ***A credential*** is a piece of information that is use to prove the identity of a subject. Passwords and certificates are examples of credentials.
- ***Authentication*** is the process by which a subject proves its identity to a requestor, typically with a credential. Authentication in which both parties (i.e., the requestor and the requested) authenticate themselves to one another simultaneously is referred to as mutual authentication.
- ***An object*** is a resource that is being protected by the security policy.

• ***Authorization*** is the process by which we determine whether a subject is allowed to access or use an object.

• ***A trust domain*** is a logical, administrative structure within which a single, consistent local security policy holds. A trust domain is a collection of both subjects and objects governed by single administration and a single security policy.

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

1. The grid environment consists of multiple trust domains. The policy element states that the grid security policy must integrate a heterogeneous collection of locally administered users and resources. In general, the grid environment will have limited or no influence over local security policy. Thus, we can neither require that local solutions be replaced, nor are we allowed to override local policy decisions.

2. Operations that are confined to a single trust domain are subject to local security policy only. No additional security operations are imposed on local operations by the grid security policy. The local security policy can be implemented by a variety of methods, including firewalls, Kerberos and SSH.

3. Both global and local subjects exist. For each trust domain, there exists a partial mapping from global to local subjects. Each user of a resource will have two names, a global name and a local name on each resource (both names may be same or different). The mapping of a global name to a local name is place-specific. The existence of the global subject enables the policy to provide single sign-on.

4. Operations between entities located in different trust domains require mutual authentication.

5. An authenticated global subject mapped into a local subject is assumed to be equivalent to being locally authenticated as that local subject. Within a trust domain, the combination of the grid authentication policy and the local mapping meets the security objective of the host domain.

6. All access control decisions are madden based on the local subject.

7. A program or process is allowed to act on behalf of a user and be delegated a subset of the user's rights. This policy element is necessary to support the execution of long-lived programs that may acquire resources dynamically without additional user interaction.

8. Processes running on the same subject within the same trust domain may share a single set of credentials.

We note that the security policy is structured so as not to require bulk privacy (i.e., encryption) for any reason. Export control laws regarding encryption technologies are complex, dynamic and vary from country to country. Consequently, these issues are best avoided as a matter of design. We also observe that the thrust of this policy is to enable the integration of diverse local security policies encountered in a computational grid environment.

GRID SECURITY ARCHITECTURE

The security policy defined in above sections provides a skeleton within which we can construct specific security architecture. In doing so, we specify the set of subjects and objects that will be under the rules of the security policy and define the protocols that will govern interactions between these subjects and objects. The most important components here are entities, credentials, and protocols.

In our focus are computational parts in grid. The subjects and objects in our architecture must include those entities from which computation is formed. A computation consists of many processes, with each process acting on behalf of a user. Thus, the subjects are users and processes. The objects in the architecture must include the wide range of resources that are available in a grid environment: computers, data repositories, networks, display devices, and other devices.

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 on behalf of a particular user. However, it is impractical for that “User A” 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 of time (i.e., days or weeks), the user may wish to allow a computation to operate without intervention. One decision is the concept of a user proxy that can act on a user’s behalf without requiring user intervention. The user proxy acts as a stand-in for the user with credentials. From management of resources point of view, we define resource proxy as an agent to translate interdomain security operations and local intradomain entity. Given a set of subjects and objects, the architecture is determined by specifying the protocols that are used.

The user proxy is a special process started by user that executes on one of Grid community member host. The main of role of user proxy protocol are credentials (passwords, private keys) given to the proxy. The proxy could use these credentials to log-on and authenticate with resource proxy. The similar scenario there is in Kerberos protocol[6].A public key system (private and public key) is used to generate temporary certificate which is used to verify that this certificate exist.

The resource proxy on the other hand is responsible for scheduling access to a resource and mapping of computational entity of resources. A user proxy requires access to resource and determines the identity of that resource proxy. If the request is successful, the resource is allocated. The inverse situation is allocation failure – the user cannot recognize user of resource (authentication failure). Another situation is when the user is not entitled to use the resource in requested mode (authorization failure). Depending on the nature of the resource and local policy, authorization may be checked at resource allocation time or process creation.

A central moment of security policy is correct mapping between global and local subjects. The resource proxy maintains a mapping table. First the user proves that he holds credentials for global and local subjects. Then he asserts a mapping between those credentials. Coordination of this process is from resource proxy. Part of mapping procedure requires the user log-on the resource for which mapping is created. This requires a user authenticate themselves to the local host. Resources with strong authentication based on Kerberos or SSL will result in amore secure mapping.

CONCLUSIONS AND FUTURE WORK

We have described requirements, policies and difficulties in building security architecture for Grids. We hope to develop these techniques and policies described above with Globus GSI and specially GSS-API. GSS-API allows to construct security policy simply by transcribing the grid security protocols into GSS calls. It’s oriented toward two party security context. It provides obtaining credentials, authentication, signing messages and encrypting messages. Mainstreams are flexible access control, secure group communication, interdomain access control policies. The flexibility of GSS-API implementation allows to switching between public key and passwords.

One interesting approach is an application of policy and practice in Active Directory in Windows enterprise platforms, - in Grid architecture. We want to implement AD policy in defining security grid architecture – management of common resources in secure domain.

REFERENCES

[1] I. Foster, C. Kesselman, S. Tuecke . The Anatomy of the GRID - Enabling Scalable Virtual Organizations ***2001***

[2] I. Foster, C. Kesselman, J. Nick, S. Tuecke. The Physiology of the GRID – www.globus.org/research/papers/ogsa.pdf

[3] I. Foster and C. Kesselman. The Globus project: A progress report. In Heterogeneous Computing Workshop, March 1998***.***

[4] K. Hickman and T. Elgamal. The SSL protocol. Internet draft, Netscape Communications Corp., June 1995.Version 3.0.

[5] J. Kohl and C. Neuman. The Kerberos network authentication service (v5). Internet RFC 1510, September1993.

[6] B. Tung, T. Ryutov, C. Neuman, G. Tsudik, B. Sommerfeld, A. Medvinsky, and M. Hur. Public key cryptography for cross-realm authentication in Kerberos.

[7] A. Vahdat, P. Eastham, C. Yoshikawa, E. Belani,T. Anderson, D. Culler, and M. Dahlin. WebOS: Operating system services for wide area applications. Technical Report UCB CSD-97-938, U.C. Berkeley, 1997.

[8] T. Ylonen, T. Kivinen, and M. Saarinen. SSH protocol architecture. Internet draft, November 1997.

[9] J. Linn. Generic security service application program interface, version 2. Internet RFC 2078, January 1997

ABOUT THE AUTHOR

Alexander Kemalov, Department of Hierarchical Systems, Institute of Computer and Communication Systems–BAS Sofia, Phone: +359 979-2774, E-mail: sasho@hsi.iccs.bas.bg.