Security Implications of Typical Grid Computing Usage Scenarios

Marty Humphrey
Computer Science Department
University of Virginia
Charlottesville, VA 22904
humphrey@cs.virginia.edu

Mary R. Thompson
Distributed Security Research Group
Lawrence Berkeley National Laboratory
Berkeley, CA 94720
MRThompson@lbl.gov

Abstract

A Computational Grid is a collection of heterogeneous computers and resources spread across multiple administrative domains with the intent of providing users easy access to these resources. There are many ways to access the resources of a Computational Grid, each with unique security requirements and implications for both the resource user and the resource provider. A comprehensive set of Grid usage scenarios is presented and analyzed with regard to security requirements such as authentication, authorization, integrity, and confidentiality. The main value of these scenarios and the associated security discussions is to provide a library of situations against which an application designer can match, thereby facilitating security-aware application use and development from the initial stages of the application design and invocation. A broader goal of these scenarios is to increase the awareness of security issues in Grid Computing.

1 Introduction

One goal of software designed as infrastructure supporting Computational Grids is to provide easy and secure access to the Grid's diverse resources. Infrastructure software such as Legion [9] and Globus [6] enable a user to identify and use the best available resource(s) irrespective of resource location and ownership. However, without an adequate understanding of the security implications of a Grid, both the Grid user and the system administrator who contributes resources to a Grid can be subject to significant compromises in security. As Grids move from an experimental phase to production facilities [14, 10, 1, 15] understanding and controlling the security of a Grid application becomes imperative.

The importance of security-related issues will amplify as Grid usage becomes more commonplace. Before a user runs an application on a particular machine, the user may need assurances that the machine has not been compromised, which could subject her proprietary application to being stolen. When a user's job executes, the job may require confidential message-passing services, which might not be the default. A user or the Grid infrastructure software may set up a long-lived service such as a specialized scheduler and require that only certain users be allowed to access the service. In each of these cases, the developer of the application must anticipate these security requirements and design the application to provide this required security-related functionality. Additionally, the invoker of these applications must understand how to check if these security services are available and how they can be invoked.

The purpose of this paper is to review the various Grid usage scenarios and analyze their security requirements and implications. *These scenarios are designed to provide guidance for the Grid user, the Grid application developer, and the Grid resource provider.* For the Grid user, these scenarios describe the security implications related to the interaction with existing components of a Grid. For the Grid application developer who wishes to design and deploy an application for use in a Grid and does not know where to begin with regard to computer security, these scenarios provide a library of cases by which to match against "best practices". For the Grid resource provider, these scenarios describe what can be expected of applications (and users) that may run on their resources, specifically with regard to interaction with other parts of the Grid and the local machine itself. In general, the intent of these scenarios and their analyses is to foster the development and deployment of interoperable security-aware Grid applications from first designs, eliminating the need to redesign and *patch* applications to accommodate the security concerns that may arise as a result of large-scale deployment and availability.

2 Preconditions to user Grid sessions

Before presenting and analyzing usage scenarios, it is important to discuss the security infrastructure that is likely

to exist in a Computational Grid independent of the daily "Grid Sessions" of the individual users. A Grid Session is roughly defined as the activities that a particular user might perform during a single workday. First, those conditions and requirements that should exist before *any* user can use the Grid are presented, followed by a discussion of the steps that a particular user must take in order to establish a Grid Session (and subsequently engage in any of the usage scenarios in Section 3).

The following assumptions are made about the Grid Computing environment as a whole:

Grid-wide Unique IDs. Each user and principal will have a Grid-wide identity that all the other Grid principals, regardless of administrative domain, can verify.

Some Resources Will Require Local IDs. Some local resource managers will require legacy local user IDs for use of their resources, so there must be a way under the control of the local administrator to map from Grid IDs to local user IDs. Similarly, access control will be enforced both by local resource managers often using legacy access control mechanisms and by Grid-aware services that may want to use Grid-centered access policies. In either case there must be simple ways for users to request access rights and allocations and the stakeholders to grant them. The issues of identity mapping are discussed in Section 4.2.

Multiple Authentication Sources. It is unlikely that all IDs will be issued and verified from a single source (even if that source is replicated). Therefore, applications must be prepared to obtain and evaluate the public statement of those conditions under which each authentication source agrees to be the Authentication Server for the entity in question. Applications must be made capable of judging the credibility of authentication servers with regard to the service they provide.

The following steps should take place prior to a particular user engaging in a Grid Session:

Allocation Requests on Per-Resource Basis. Some sites (such as supercomputer centers) may require that each individual have a local user ID and allocation, while other sites may allow group allocations or simply require that a Grid user be permitted to use the resource possibly in a constrained manner (e.g. only on weekends or late nights). Establishing permissions and allocations on a resource depends on the resource owner's policy and may require sending email to the system administrator of the resource in question (perhaps via a Web interface).

Short-lived Credentials. The use of short-term proxy certificates in place of the long term Grid ID (i.e., private key) is a desirable feature of a distributed system, since it limits the exposure of long-term private keys.

Per-Session Security Parameters. While many security sessions are set just for the duration of a particular activity on the Grid, a person may wish to establish security parameters that exist for the life of the session. For example, a person may specify a specific role that she wants to assume, such as system administrator for a particular resource or ordinary user.

3 Usage scenarios

The scenarios are summarized into six categories: immediate job execution, job execution that requires advance scheduling, job control, accessing grid information services, setting or querying security parameters, and auditing use of Grid resources. The unique security implications of each group of scenarios are discussed in turn. In these scenarios the term *Grid user* or *user* refers to the person who is attempting to access a resource; *principal* is used to mean any entity, either human or process that has an identity associated with it and wants to make use of or to provide resources; *stakeholders* are people or organizations who set the use policy for a resource; a *Grid gateway* is a process which accepts remote requests to use resources; a *Grid resource gateway* is the process that actually controls the use of the resource (this may be legacy code); a *Grid administrator* is a Grid-aware person with responsibility for the overall functioning of the Grid (note that there will probably exist multiple Grid administrators with non-overlapping realms of responsibility in a single Grid); and *site administrators* are responsible for the functioning of a single site. The *user's home organization* is the administrative domain to which the user belongs which may have trust relationships or service agreements with some of the resource providers.

3.1 Immediate job execution

The first scenario analyzed is that of a user who wishes to combine resources from multiple sites into a single, coordinated job for immediate execution. For example, a user could generate a large amount of data from a major shared instrument (e.g., accelerator or microscope study), which then needs to be uploaded to a large data store that in turn can be accessed by a powerful compute engine. Once preliminary data analysis has taken place, intermediate data may need to be saved and also passed on to a different compute engine for further analysis such as visualization procedures.

The specific resource sites may be selected by an agent acting on behalf of the user based on user defined metrics such as "quickest", "availability" and "cheapest". The

choice is made by a third-party service, such as one of the emerging "super schedulers" as exemplified by the default scheduler in Legion. The user may specify a group of resources from which to choose, or the user may leave it to the super scheduler to locate the set from which to choose. Remote job execution, especially at multiple sites, is likely to require both reading and writing of files from remote sites. The security requirements of such a scenario include:

1. If the set of candidate hosts has not been identified by the user, the super scheduler will need to interact with the Information Services component(s) of the Grid to identify the set of possible hosts.

2. The super scheduler must determine if the target user is allowed to execute on each of the target Grid machines, and, if so, the remaining allocations of the user. This information is determined from Information Services or querying each Grid machine directly.

3. A controlling agent or each remote job in a sequence needs to request resources on behalf of the user, perhaps through subsequent calls to a super scheduler.

4. Mutual authentication of user and Grid gateway on specified host needs to be done before a piece of the job is run there.

5. The grid gateway on a specified host must map the Grid ID to a local ID and submit the request to the resource gateway so that the job will run as the authorized local user.

6. The executing jobs may need to be given authorization to read and write remote files on behalf of the user.

7. If the remote job writes output to files on an AFS or DFS file server, it needs the user's Kerberos ticket (which may or may not be the same as the credentials used to authenticate to the Grid gateway).

The super scheduler, the controlling agent and each remote job that needs to read or write files must be able to act on the user's behalf. The super scheduler needs to make inquiries as to machine characteristics and availability. Site-wide detailed information about machine and account information is largely regarded as being important to keep secret, so it will probably be the case that an arbitrary entity will not be allowed to query it. Therefore, either the super scheduler, as a principal, must be granted broad access to such information and trusted not to leak such information to any one except the affected user, or the super scheduler must be explicitly granted the right to ask on behalf of the user.

Authorization to use the target machine is performed by a Grid gateway server. When a job involves a sequence of processes run on different hosts and domains, there is either a controlling agent that starts the jobs in sequence or else each process must be able to start the next piece. In either case some entity other than the user will be asking a Grid gateway to start a job. This entity must be able to present a credential that will grant it the same privileges as the user. In the case where the running process needs to do remote file I/O or start another remote process, it too, will need a credential to act of behalf of the user. See Section 4 for a discussion of the challenges of credential delegation.

3.2 Job execution requiring advance scheduling

If the large data flow from an instrument must be processed in real time, it may require the advance reservation (or *co-scheduling)* of data storage, network bandwidth and possibly compute cycles. Advance reservations require:

1. Delegation of the user's rights to a super scheduler and bandwidth broker to make the reservations on behalf of the user.

2. Assurance that if a user has been granted a reservation for the future, she will have access at the time the reservation is claimed.

3. Bandwidth reservations usually require service agreements for priority bandwidth between ISP's and compute sites. This implies that a bandwidth broker needs to know at reservation time that user's connection will come from an authorized site.

If the model of execution is such that the bandwidth broker returns a claim ticket to the super scheduler, the transfer of the claim ticket from the super scheduler to the user must be protected, and the claim ticket itself must be non-forgeable. When the job is going to be run, it needs to be able to claim the reservation. The execution of a job on reserved resources can require multiple concurrent claiming procedures. In this model, a user directly interacts with the individual resource gateways to claim the reservation. In general, reservation claiming requires:

1. The user must be able to identify himself as the entity that made the reservation. The reservation may have been made on behalf of a group, in which case the user has to prove himself to be a group member. Another way of handling the situation where one person makes a reservation and a different person wants to claim it, is to allow the claim tickets to be transferred. In this case the resource gateway must be able to verify that the claim has been legitimately transferred by the person who made the reservation to the current claimant.

2. The user should still have access to all the resources that he has reserved, except in extreme cases, such as

when the user is no longer associated with the organization that is going to pay for the resource use or the organization has failed to pay its bills.

3. In the case of a user losing access to a resource, a check should be made of advance reservations in his name, and the appropriate parties should be notified of the change.

This scenario contains two important requirements in Grid Computing, *group membership* and *nonrepudiation*. Group membership is non-trivial because, while individual users should be able to define groups, consensus has not been reached regarding how exactly to do this. Nonrepudiation in this context refers to the requirement that the resource gateway should not be able to arbitrarily deny that it granted a reservation.

3.3 Job control

A standard requirement of users with long-running remote jobs is the ability to disconnect from a job and then at a later time and possibly from a different location reattach to it. The user may just want to monitor the progress of a job, or may want to enter some steering information at specific points in the run. Monitoring a job's progress may be as simple as knowing where logging files are being written and having the access to read them. Steering implies that the user has defined *entry points* into the computation and has some way of controlling who may connect to them. In the collaborative environment facilitated by the Grid, a different user may want to use the monitoring or attachment points as well. The user in this scenario would probably rely on pre-defined libraries generated by security developers rather than creating an individual security solution. Utilizing well-accepted libraries facilitates interoperability. A second sort of job control can occur when a job appears to the system administrator to be out-of-control and should be forcibly terminated.

1. In this case the resource that is being protected is access to a running job created by a user, who will set the access policy and later be granted access by that policy. This can perhaps be most easily accomplished if the policy and code to enforce access is part of the job.

2. The point of entry is probably directly to the computation itself as opposed to through the Grid gateway or the resource gateway, so the potential collaborator must be able to authenticate to the computation itself.

3. In the case of a forced termination, the system administrator must detect the out-of-control process and trace its origin to a particular Grid user. Alternatively, Grid monitoring software might detect the out-of-control process and notify the system administrator.

4. The system administrator should be able to inform the Grid Administrators that the process is about to be terminated. The Grid Administrators need this information to coordinate the termination of this job across multiple Grid sites.

5. The Grid Administrators either attempt to terminate the individual components of the job by directly interacting with the job or by asking the system administrators to terminate those processes of the job that are on their respective machines.

6. The job owner must be notified by the Grid Administrator that his job has been terminated.

The job here is considered a "resource" to which the user who started it, and the system administrator have certain default rights. Since resources of a Grid are used both by "local" users and Grid users, it is not necessarily obvious from where a process originated. Therefore, Grid software must keep audit records or at least provide a means by which local jobs can be identified with the Grid user who started them. In the case of forced termination, there will generally not be a single person who has the power by which to kill a typical Grid Computation, because it will span multiple administrative domains. As such, ideally, a coordinated effort must be made if a single job is to be prematurely terminated (note that this is unlikely at least in the near term). Finally, the user must be told at the very least that her job has been prematurely terminated, as opposed to the computation just disappearing.

3.4 Accessing grid information services

The ability to locate services and to determine the status and availability of those services will be crucial in a well-functioning Grid. In most Grid architectures, there exist Information Services whose purpose is to be a centralized repository for such information. Many services require carefully controlled access to information regarding the services they provide, their current status, and who can use them. Users will mostly be reading from the Directory Service but entities such as machines and monitoring process will want to enter information and set access policies for their information. In general, when a Grid user queries or updates an information server:

1. Authentication should take place between the user and the information services.

2. The information services should implement the access control policy as desired by the service.

3. When publishing information, confidentiality or message integrity on the communication from the pub-

lisher to the information services could be required by the publisher.

While the information services require the user to authenticate, it is not strictly necessary for information services to authenticate to a reader. For example, if the user subsequently authenticates to the service itself, that will validate the information he received. If there are multiple Directory Services that provide the same information, the user may require server authentication to help decide the value of possibly conflicting information. The extra cost of mutual authentication in general can be weighed against the potential effects of malicious information.

With regard to the information services providing the actual information requested, it could be the case that the individual services are allowing the information services to determine an *appropriate* access policy. However, a more general scenario is to allow each publisher to set the policy. In this case, the publisher and the information services must agree on a policy language. Subsequently, the publisher must trust that the information services accurately implements the policy.

3.5 Setting or querying security parameters

There is a large number of parameters that affect the security of a user's interaction with Grid services and resources. These need to be set by both the user and the stakeholders. The integrity and confidentiality of both messages and stored data are examples of such parameters. Integrity refers to the property that data cannot be noticeably altered between when it was written and when it is read. A user might want to specify which MAC algorithm, if any, is used to ensure integrity. Confidentiality means that no one aside from the writer and the intended reader(s) can understand the data. The parameters to set here are the encryption method and strength and lifetime of keys. If both the stakeholder and the user specify these parameters, the supporting software must be able to negotiate to find a solution that is acceptable to both. Typically the way this is done in Transport Layer Security (TLS) [2] style software is for both parties to specify a set of acceptable parameters.

1. Data integrity implies supporting MAC algorithms.
2. Confidentiality requires supporting a key agreement protocol.
3. Services and applications must recognize the rationale for per-user security configurability and be designed accordingly.
4. There must be a secure and efficient mechanism to negotiate a particular user's integrity and confidentiality parameters with those of the service.
5. The long term storage of encrypted data requires the user and/or server to have long-term storage and escrow of encryption keys.
6. The server that is writing the file to storage may need to share an encryption key with the owner of the file.

This scenario exemplifies one of the key challenges in constructing a Grid—namely that there is a tension between support for heterogeneity and a requirement that services implement some subset of shared functionality. Many stakeholders will implement and deploy services for a Grid, each with a different API and different functionality. However, their utility will be significantly impeded if they do not provide flexible user interfaces. Requirements for message integrity or confidentiality is an example of a requirement that may be imposed across a class of applications from their perspective users.

In general, proper key management is a requirement for many of the scenarios. For example, certain administrative domains within a Grid may require smart cards for key management, as opposed to a password-based authentication scheme. The requirements for key management must be properly conveyed to the users by the Grid Administrators. Managing keys will be a challenge for the user, as a Grid may cross multiple administrative domains.

Another broad category of security parameters is the authorization policies for each resource. In this example it is assumed that there is an authorization policy interpreter that can be queried. A user may need to determine his own access to a resource before attempting to use it. A stakeholder or scheduling agent may need to know another's access rights with respect to a resource. A stakeholder for a resource on a remote machine may want to set or modify the policy for the resource's use. A stakeholder may need to quickly revoke access to a user or set of users. The implications of these requirements are:

1. Either the resource gateway or an independent policy analyzer must be able to determine a user's access given the Grid ID of the user and decide if the principal asking the question has the right to see the answer.
2. For policy information stored on the resource gateway or by an independent authorization server, the stakeholder must be able to connect in a secure and authenticated way to the gateway (and subsequently edit a policy file) or authenticate himself to a server that can modify the policy information.
3. If the policy information can be stored locally to the stakeholder, the authorization policy must be digitally signed and kept securely.

4. Policy information may need a validity period or a priority assigned to it if the policy is intended to be temporary.

5. Any caching of access rights must be short-lived and/or provide a way of being flushed.

6. If policy information is stored in distributed places or multiple copies are kept, it must be linked together or indexed in some way so that all the copies can be deleted.

7. If capabilities are used they must be very short-lived or else kept in known places from which they can be removed.

A challenge in supporting stakeholder defined access policy is that there may be multiple stakeholders that have jurisdiction over different usage rights of a single resource. Therefore, the server that maintains the policy must carefully enforce the policy regarding each stakeholder's ability to change the access policy.

Another category of security parameters is the trust relationships between users and administrative domains or hosts. As part of a session-specific configuration or in a directed scheduling request, a user may want to specify what hosts she is willing to use. If a job is going to use several hosts this information has to be passed along to the scheduler or the job controller. Similarly, a service provider may mandate that requests for service must arrive from a particular subset of hosts, perhaps because the other hosts are not trusted or because of billing considerations. Lastly, a Grid administrator may specify that no user or service is allowed to interact with users or services from another administrative domain. For example, if NASA trusts DoD, but DoD does not trust NASA, then the DoD Grid Administrator(s) might require that DoD users cannot use NASA machines in DoD-related computations. To support the specification of trusted Grid hosts or trusted Grid domains:

1. Grid hosts must be able to authenticate and possibly prove membership in a particular Grid domain. This can be done through host SSL credentials or secure DNS and IPSEC.

2. Servers in this category require a protocol in which both the identity and location/domain from which the request originated are authenticated. Clients must be ready to provide this information.

3. Grid administrators must be able to enforce these requirements.

Implementation of this requirement can be problematic with regard to all entities that could specify a set of trusted Grid hosts. For example, if the computation scenario is such that there is a chaining of services (e.g., user asks server$_1$, server$_1$ asks server$_2$, server$_2$ asks server$_3$, ., server$_n$ returns information back to the user), then the entire chain might be required to be authenticated before server$_n$ performs the requested action and subsequently returns the information to the user. Similarly, server$_{n-1}$ must know the user's restricted set of hosts before contacting server$_n$. This is not easy for the Grid software to enforce.

3.6 Auditing use of Grid resources

Either a site system administrator or a Grid administrator may need to monitor all accesses to the resources at a site, or the stakeholder may want to monitor the use of just his resource. This information may be used for accounting purposes, for a routine security review, or for a real-time intrusion detection procedure. The system administrator may wish to check both the accesses allowed and the accesses rejected. This scenario implies:

1. The resource gateway server must keep an non-forgeable log of all access by unique user identification and time of access.

2. The format of the entries to this log must be negotiated between the system administrator and the resource gateway.

3. Access to this log should be carefully restricted, but stakeholders need to be able to see the entries for their resources.

4. There is a need to identify a stakeholder with a resource.

5. To accomplish real-time intrusion detection, the resource gateway needs recognize and signal especially troublesome resource access requests in additions to logging.

4 General security issues and challenges

The Grid security requirements can be grouped into several broad categories each with its own challenges.

4.1 Delegation

Many different usage scenarios require one agent to act on behalf of a principal. The conventional approach when a user must ask a service to perform some operation on her behalf is to grant unlimited delegation, which is to unconditionally grant the service the ability to impersonate the user. While this is a reasonable approach in an environment in which all services can be wholly trusted by the users who

wish to invoke them (and is the current state of the art), it is clearly not scalable into general-purpose Computational Grids. For all delegations that occur in a Grid, the crucial issue is the determination of those rights that should be granted by the user to the service and the circumstances under which those rights are valid. Clearly, delegating too many rights could lead to abuse, while delegating too few rights could prevent the task from being completed. To date, restricted delegation is not used in emerging Grids because it is difficult to design, implement and validate except in very limited, ad hoc cases. Some of the challenges are:

1. Knowing the minimum set of rights that the execution of a job requires. One of the problems is in how rights are named by various servers.

2. Knowing how many levels of delegation are required. If the user is using code that he did not write he will not know how many servers may be called in accomplishing the task. Even in well known code each job may require access to different sets of servers.

3. When a resource gateway receives a chain of delegated certificates, it must decide whether to trust all the intermediaries that the delegation has gone through. This may require rather large, open-ended trust relationship policies on the part of the gateways. The exact delegation of the users rights may not be under the direct control of the user, and the user may be unaware of the trust relationships of all the hosts in the system. Thus a legitimate request from an authorized and trusted user might arrive at a destination and be rejected because it had passed through an untrusted domain.

Recent work has begun to more carefully establish the dimensions along which we would like to restrict delegations [18]. These include:

1. Specify the rights that may be delegated.

2. Specify a limited time period during which the delegated credential is valid. The problem with this is knowing how long a job will take.

3. Specify to what principals (servers or users) the rights may be delegated. Again, knowing the complete set of servers that may be invoked in job execution is problematic.

The GSI subgroup of the Security working group of the Grid Forum [8] is also investigating restricted delegation.

4.2 Identity mapping

Mapping Grid identities to local user IDs is a way to enable a user have a single Grid sign-on and yet support legacy access control mechanisms on those sites that require it. This implies that a user must have a local ID at the sites that require one, and that the site administrator and the Grid administrator agree on the mapping to be used by the Grid gateway server.There are several security implications raised by this model: it requires users to have local accounts on any machine they want to use; it may give the user more access to the host than he needs, for example he may be able to run many applications rather than those explicitly specified by the gatekeeper; it requires the Grid administrators to trust the host's access control and accounting procedures, and the local site to trust Grid CA's to correctly identify users, and the Grid software to authenticate them.

On the other hand, many existing compute centers require that a user has an account with them and then rely on the underlying OS to do authorization based on the userid. Both the Globus and Legion middleware support such mapping files.

A mechanism for allowing the local administrator to specify trust relations with various CA's and other sites could be used rather than a direct mapping of ID's. For example, an administrator might be willing to allow a user signed for by a given CA to run as a trusted user.

4.3 Grid information services

Most Grid environments will support an information service to allow potential users to locate resources and to query them about access and availability. In general, sites are unwilling to allow unrestricted access to such detailed information about their sites. Thus, access to this information will be controlled. Current directory services are implemented using the LDAP protocol which has its own user/password based access control. A mechanism is needed to either use Grid credentials as the basis for directory service access control or to map the user's Grid ID to a directory service user name.

4.4 Firewalls and virtual private networks

Firewalls or VPN's between the user's host and the server host, or between different server hosts present a serious challenge to Grid security measures. Grids that span administrative sites and encourage the dynamic addition of resources are not likely to benefit from the security that static, centrally administered commercial firewalls or VPN's provide. On the contrary, Grids need to enforce their own security and a firewall is likely to prevent Grid-authorized accesses. Typically firewalls only allow access from or to specific hosts and to specific ports. The Grid infrastructure servers can be configured to run on known ports which can be allowed by the firewalls. User provided servers and code tend to be more unpredictable in their port usage and it may

not be possible to run them on hosts that are behind firewalls. Also jobs that are scheduled to run on the "best" set of hosts may break if the request does not arrive from an allowed host.

VPN's usually require some specific authentication and authorization in order to make a connection. Some VPN's support x509 identity certificates for authorization and might be able to use Grid IDs. Such a VPN might present a way to get through firewalls and allow the standard Grid access control to work.

4.5 Related work

Several Grid and Collaboratory projects have done similar surveys of security requirements. The papers that are closest in scope to this paper are the RFC's issued by the Authorization Accounting Architecture Research Group [19, 4]. The first of these papers lists 6 network applications and the authorization they require. The example they give that is closest to a Grid application is a Network bandwidth broker which is similar to our super scheduler scenario. Their other applications include mobile IP, distance learning, electronic commerce. The second paper gives a list of requirements for an AAA protocol that can support such applications. They have the following high level requirements: 1) Authorization decisions must be made on the basis of information about the user, the service requested and the operating environment. Information about the user must include extensible attributes as well as identity. Unknown users must be supported. 2) Identity and attribute information must be passed with integrity, confidentiality, and non-repudiation. 3) Authorization information must be timely (and revokable) 4) Support application proxying for users, 5) support ways of expressing trust models between domains 6) Protocol must support context sensitive decisions as well as transactions, 7) Both centralized and distributed administration of authorization information. 8) Separate or combined messages for authentication and authorization 9) Authorization information should be usable by applications, including accounting and auditing applications. 10) Support negotiation of security parameters between requestor and service. Since we are not currently specifying a protocol, some of these items are not part of our goals, but we agree on the general need for authorization based on user identity and attributes, the need for proxying users, the need to support ways for stakeholders to set use policy, ways to define trust between domains, and the need for the service providers to negotiate security parameters with the users.

Johnston, et. al. [13] have also written about the special security considerations of Grids based on the experience of the NASA Production IPG grid as well as experience with several DOE collaboratories. They considered the threat model and risk reduction in greater detail than our paper and came up with a security model based on using available Grid security services.

Both Globus and Legion have published several papers about their security models. Globus emphasizes the need for a single sign-on for users, protection of user credentials (passwords, private keys, etc.), interoperability with local security solutions, uniform credentials/certification infrastructure [7]. The Legion security papers also identify requirements and approaches for Grid computing from the perspective of object-based computing. They identify the following: Isolation of nodes, so that a compromise of one node will not affect other nodes; detection and recovery from security breaches; access control for resources; communication privacy and integrity; Grid-wide identity of principals; a flexible authorization infrastructure that supports CA and CA-less configurations; and integration with standard mechanisms such as Kerberos, DCE and ssh to satisfy local policy and legacy applications [5]. They also provide a detailed analysis of the roles and potential threats resulting from different configurations of the gatekeeper entity [11]. The DOE supported Diesel Combustion Collaboratory which was tasked with providing a secure collaboration environment did a survey of collaboratory security needs [17]. They also identified the need for a common user identity to support single sign-on and the need for delegated proxies for remote computations involving several resources. In addition they specified some needs directly related to collaboration between users such as secure e-mail and video and audio conferencing.

5 Conclusions

Computational Grids are rapidly emerging as a practical means by which to perform new science and develop new applications. The goal of this paper was not to discuss the particular security mechanisms or policies of systems such as Legion, Globus, or any other existing system, but rather to describe Grid security that transcends existing approaches. Each scenario in this paper is designed to provide guidance for the Grid user, the Grid application developer, and the Grid resource provider. While a given scenario can provide practical guidance for design and deployment, additional insight is gained by recognizing the general, rapidly-emerging issues such as the need for restricted delegation (giving only a subset of your rights to something that will act on your behalf) that can be seen running through many of the scenarios.

There are many subtle security implications involved in the many emerging Grid usage scenarios. Both the resource provider and the resource consumer should understand, from a security perspective, what is expected from each other and what might happen if these expectations are

not met. Without this understanding, the transition from experimental systems into production systems will soon be curtailed by explicit security violations or more subtly a compromise of information that a user had believed was securely kept private.

Acknowledgments

We are grateful to the many members of the Grid Forum Security working group who contributed to the discussions regarding this topic at Grid Forum 4 in Seattle in July 2000. In particular, Steve Tuecke greatly aided the organization of these topics. Many members of the Scheduling working group also contributed to the development of these ideas. In particular, Keith Jackson read versions of this draft and enhanced its development.

Marty Humphrey was supported in part by the National Science Foundation grant EIA-9974968, DoD/Logicon contract 979103 (DAHC94-96-C-0008), and by the NASA Information Power Grid program. Mary Thompson was supported in part by the U.S. Dept. of Energy, Office of Science, Office of Advanced Scientific Computing Research, Mathematical, Information, and Computational Sciences Division under contract DE-AC03-76SF00098. Its report number is LBNL-47047.

References

[1] Department of Energy Science Grid, http://www.itg.lbl.gov/Grid

[2] T. Dierks, C. Allen. The TLS Protocol - Version 1.0. IETF RFC 2246 Jan 1999 work-in- progress.

[3] European EGrid, http://www.egrid.org

[4] S. Farrell, J. Vollbrecht, P. Calhoun, L. Gommans, G. Gross, B. DB Bruijn, C. DB Laat, M. Holdrege, D. Spence. AAA Authorization Requirements. RFC 2906, Informational, work-in-progress, August 2000.

[5] A. Ferrari, F. Knabe, M. Humphrey, S. Chapin, and A. Grimshaw. A Flexible Security System for Metacomputing Environments. *Proc. High Performance Computing and Networking Europe 1999*, Amsterdam, April 1999.

[6] I. Foster and C. Kesselman. Globus: a metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications*, 11(2):115-128, 1997.

[7] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. A Security Architecture for Computational Grids. *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pg. 83-92, 1998.

[8] Grid Forum, http://www.gridforum.org/

[9] A. Grimshaw, W. A. Wulf, *et. al.*. The Legion Vision of a Worldwide Virtual Machine. *Communications of the ACM*, 40(1):39-45, January 1997.

[10] High Energy Physics Data Grid, http://les.home.cern.ch/les/grid/welcome.html

[11] M. Humphrey, F. Knabe, A. Ferrari, and A. Grimshaw. Accountability and Control of Process Creation in Metasystems. In *Proceedings of the 2000 Network and Distributed Systems Security Conference (NDSS'00)*, San Diego, CA, February 2000, pp. 209–220.

[12] M. Humphrey, M. Thompson. Security Implications of Typical Grid Computing Usage Scenarios, October 2000 Informational Draft, http://www.gridforum.org/security/drafts/draft- gridforum-security-implications-01.pdf.

[13] W. E. Johnston, K. Jackson, S. Talwar. Security Considerations for Computational and Data Grids. *10th IEEE Symposium on High Performance Distributed Computing (poster session)*, Aug. 2001.

[14] NASA's Information Power Grid, http://www.ipg.nasa.gov/

[15] National Partnership for Advanced Computational Infrastructure (NPACI) Legion Network, /http://legion.virginia.edu/npacinet.html

[16] B. C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications Magazine*, 32(9):33-38, September 1994.

[17] C. M. Pancerell, L. A. Rahn, and C. L. Yang. The Diesel Combustion Collaboratory: Combustion Researchers Collaborating over the Internet. *Proceedings of SC 99*, November 13-19, 1999, Portland, Oregon.

[18] G. Stoker, B. White. E. Stackpole, T.J. Highley, and M. Humphrey. Toward Realizable Restricted Delegation in Computational Grids. In *Proceedings of the International Conference on High Performance Computing and Networking Europe (HPCN Europe 2001)*, Amsterdam, Netherlands, June 2001.

[19] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence. AAA Authorization Application Examples. RFC 2905, Informational, work-in-progress, August 2000.