Scoping Security Issues for Interactive Grids

Jeffrey Dwoskin[*], Sujoy Basu[+], Vanish Talwar[+], Raj Kumar[+], Fred Kitson[+] and Ruby Lee[*]

[*] Department of Electrical Engineering
Princeton University
Princeton, NJ 08544
{jdwoskin, rblee}@ee.princeton.edu

[+] HP Laboratories
1501 Page Mill Road,
Palo Alto, CA 94304
{sujoy.basu, vanish.talwar, raj.kumar, fred.kitson}@hp.com

***Abstract*-Grid computing allows flexible resource sharing among geographically distributed computing resources in multiple administrative domains. Virtualization of resources allows jobs to be run on remote resources participating in a Grid. While this computing paradigm has been used primarily for batch jobs, we study interactive Grid applications rich in graphics and multimedia such as scientific visualization and digital content creation. A host of security issues need to be addressed for such Interactive Grids to gain acceptance, particularly in industry. The purpose of this paper is to scope these security issues.**

The Grid Security Infrastructure (GSI), a component of the Globus Toolkit [1], creates Grid credentials for every user and resource. We describe how this may be extended to securely set up an interactive session on a remote host, and the additional security issues associated with interactive session management. We propose controlled shell and controlled desktop mechanisms that restrict the user to execute only authorized commands and applications, and controlled user and super-user accounts that customize the shell and desktop using policy files. We also propose a new approach to scoping the security needs of Grid systems by defining three generic scenarios: mutual trust, partial trust and mutual distrust. New security issues arise when the user may not be trusted, or the user and the host computer's owner are mutually suspicious.

I. Introduction

Grid computing [2] [3] [4] allows computing, storage and other resources that are geographically distributed and belong to different administrative domains to participate in a virtual organization. Resources are virtualized so that members of the virtual organization can execute their applications on coordinated resources obtained by specifying the requirements, rather than identifying the individual resources to be used. Traditionally, Grid computing has been used as a mechanism for obtaining computing cycles on machines ranging from high-end computing platforms to clusters of workstations. Applications have come from various scientific computing domains, and have been submitted primarily as batch jobs. Efforts on providing security in grid and distributed computing environments are described in [5] [6] [7] [8] [9] [10] [11] [12] [13] [14].

In this paper, we present a procedural description of the security mechanisms involved in setting up and using a Grid system. We consider the additional security implications of interactive Grid applications. We also propose a new approach to scoping Grid security issues by considering three scenarios with decreasing levels of trust: mutual trust, partial trust and mutual distrust. We define a mutually trusting environment as one where the user trusts the resource provider to keep his data and activities secret and not interfere; and the resource provider trusts the user not to act maliciously. We define a partial trust (or distrusted user) environment as one where the user trusts the resource provider as before, but the resource provider places additional restrictions on the user's actions. Finally, in a mutually distrusting environment, the user also may not trust that the resource provider will always act appropriately with his data and processes.

In section II, we describe the security mechanisms associated with the current Grid architecture. In section III, we describe Interactive Grid systems and introduce their additional security needs. In section IV, we discuss the potential vulnerabilities of Grid systems in the three scenarios of mutual trust, partial trust and mutual distrust. We conclude in section V.

II. GRID Security Mechanisms

The Grid security architecture is described in [5] and the subsequent implementation of the Grid Security Infrastructure (GSI) with Globus in [6]. The security architecture for Open Grid Services is described in [7] [8]. Here, we describe key Grid security concepts and provide a procedural description of the security mechanisms underlying the setup and use of a Grid system.

The security in current Grid systems is designed to protect a legitimate user and a legitimate host system from a potentially malicious third party. Thus the focus is on authentication, authorization, and automation. Authentication will identify each entity and ensure that no third party is involved. Authorization ensures that the user is allowed to use the remote Grid resources. Finally, automation is used to implement many of the design goals of Grid computing, such as single sign-on, virtual organizations, and interaction among multiple administrative domains.

A. Authentication

Every entity, such as a user or a resource, is identified by a standard X.509v3 [15] certificate. The entity generates a public/private key pair to be used as its credentials. The private key is usually secured by a password and kept secret, while the public key is stored in a certificate. A certificate authority (CA) is a trusted third party that attests to the information in a certificate by signing it with its own private key. Before signing, the CA will typically check some form of physical identi-

 Published in the *Proceedings of the 37th Asilomar Conference on Signals, Systems and Computers*, November 9-12, 2003. Pacific Grove, CA USA.

fication to validate the information. Its signature becomes part of the entity's certificate, which can now be verified, using the CA's public key, by anyone who trusts the CA. Appendix A further describes this authentication process.

B. Certificate Authorities and Virtual Organizations

When setting up a Grid [16], each entity (user, computer/host, or other resource) must first set up a certificate. The certificate will include information such as when it expires, a serial number, which CA signed it, the entity's public key, and a subject. The subject uniquely identifies the entity on the Grid, and is composed of an organizational structure and a common name (see Fig. 1). For example, the following user subject,

O=MyOrganization, OU=MyGroup, CN=Alice

indicates the virtual organization of `MyOrganization`. Within `MyOrganization`, the user is in the organizational unit of `MyGroup`. Finally, the common name is `Alice`, which in this case belongs to a user. Similarly, a host might have a subject such as:

O=MyOrganization, OU=MyGroup,
CN=host/somehost.myorg.com

Here, the common name indicates a host instead of a user and the hostname is `somehost.myorg.com`.

Each virtual organization may run its own CA. This CA will sign all of the certificates of its members, verifying the information contained in those certificates for other entities. Alternatively a centralized or a commercial CA can be used. Every entity must maintain a list of CAs that it trusts. For each CA, it stores a copy of the CA's certificate, which contains a public key. It also stores what it trusts that CA to do. For example, a CA for MyOrganization may be trusted only for the certificates with subjects starting with `O=MyOrganization` [17].

C. Authorization and Local Accounts

After certificates have been created for each entity, the resources need to be configured. Before a user can access a host, a local account must be created or allocated for their use. A mapping must then be created from their global identification (the user's subject) to the local account. This is done in the Grid map-file. The user's job will later be managed by the local security policy using this local account. If the user will request access to multiple resources, local accounts must be acquired, and mappings created on each resource. The user may also want to access a virtualized resource instead of a specific resource. Local accounts and mappings must be acquired on each resource that might be used.

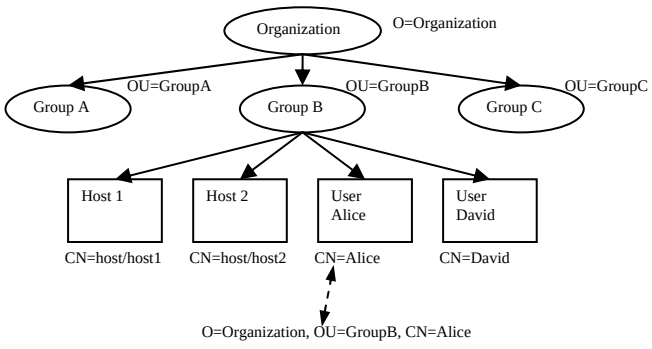

Fig. 1. Subject identifiers for entities in a virtual organization

The scalability and manageability of the system are enhanced if users are not required to have personal local accounts on each machine they use in the Grid. Dynamic template accounts [18] [19] [20], are local accounts created on the resource without a fixed user associated with it. A system administrator maintains pools of dynamic template accounts. One of these accounts is dynamically assigned to a user for running a job on the resource. At the end of the job, the dynamic account is returned back to the pool. There could be classes of dynamic accounts defined based on duties users perform, for example in their roles as scientists, design engineers, or financial analysts. Security policies need to be defined for each class of dynamic accounts. The user is mapped to one of these accounts using his access control credentials.

D. Automation and Running a Job on the Grid

Fig. 2 shows the steps involved when a user requests a batch job to be run on a remote GRID system.

One goal of automation in a Grid system is for users to have a single sign-on, rather than having to log-on and supply a password to each resource used. This is achieved by means of a user proxy (step 1). The user proxy is given a new certificate with its own key pair. The user signs the user proxy's certificate with his private key, allowing the user proxy to act on his behalf while the certificate is valid, usually 24 hours. In addition to the time limitation, the user may put further restrictions on what the user proxy can do by listing them inside the certificate. In order for the user to access his private key to sign the user proxy, he will have to enter his password, however this should be the only time the password is required, since the proxy's private key is not encrypted with any password. Thus, single sign-on is achieved.

Once the user proxy has been created, the user can launch the job he wants to run on the Grid. From this point forward, the Grid software will do all of the work with the user proxy acting on behalf of the user. The user proxy now connects to the Grid gatekeeper of the remote site (step 2). The gatekeeper is a process on each Grid site that listens for incoming requests from users and sends them to the appropriate resource. First, the gatekeeper and the user proxy perform mutual authentication. They challenge each other using a nonce, as described in Appendix A. The user proxy then checks the host's certificate so it can trust the host and its gatekeeper. Similarly the gatekeeper checks the user proxy's certificate; however, the user and not a CA signed its certificate. Therefore the gatekeeper first checks the user proxy's certificate against the user's certificate. Then it checks the user's certificate against the CA, which it trusts. After mutual authentication, both the user and gatekeeper can be sure they are talking to each other and not a malicious third party.

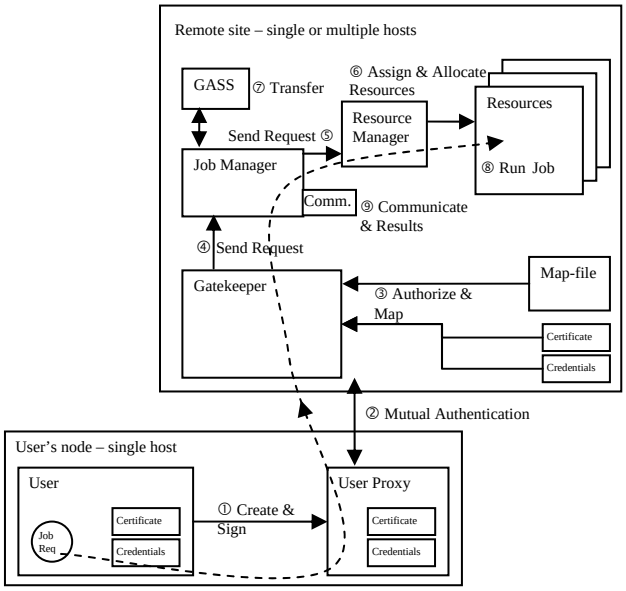

Fig. 2. Running a job on the Grid

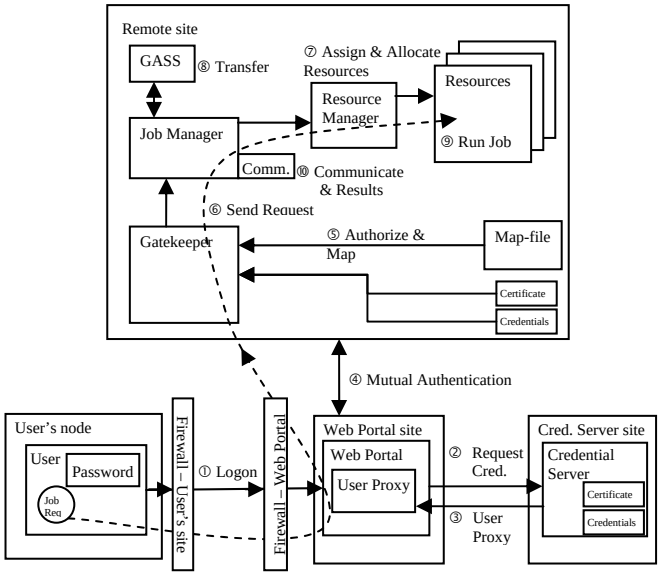

Fig. 3. Accessing Grid Using a Web Portal

Next the gatekeeper decides whether or not the user is allowed to use its services. This is determined by an entry in the Grid map file with the user's Grid subject (step 3). Once the user is authorized, the gatekeeper uses the entry in the map file to determine which local account the user will be mapped to.

After the user is authorized and mapped to a local user, the gatekeeper sends the user's request to the job manager (step 4). The job manager submits the user's request to the resource manager (step 5). The resource manager will determine which local resources are available to fulfill the user's request and will allocate them for the user (step 6). The job manager is responsible for all future communication with the user. The job manager will send the user a contact string that can be used to check on the status of the job or to cancel it in progress.

The job manager allows the user proxy to upload data or executables that are needed. This is done with a facility called Global Access to Secondary Storage (GASS). If GASS is used, the job manager contacts the GASS server on the user's node and gets the files before starting the job (step 7). GASS is also used to keep track of the output of the user's jobs. When all of the data and executables are present on the resources of the remote site, the job manager submits the job to the resource manager to start the job for the user (step 8). When it completes, the output is sent back to the user (step 9).

E. Accessing Grid Using a Web Portal

The above standard process of accessing a Grid assumes that the user is on a computer with access to Grid software and his credentials. The architecture is often modified to include a web portal and credential proxy server (see Fig. 3). An example of a credential proxy server is MyProxy server [21]. The user's computer no longer contains the user's credentials and certificate. The first step is for the user to connect to a web portal that provides the Grid software. The portal is a website accessible using HTTP or HTTPS (secure HTTP). Even though the user may have a firewall present on the edge of his network, interactions with Grid portal can still pass through it because firewalls typically allow web browsing through HTTP and HTTPS. A similar firewall permitting HTTP and HTTPS traffic could also be present on the edge of the web portal's network.

When accessing the portal, the user will not have his credentials available to authenticate or create a user proxy. Instead, the user's credentials are stored on a special credential proxy server accessible by the web portal. The user must delegate a set of time-limited proxy credentials to the credential server ahead of time. When the user logs into the web portal, it sends a request to the credential server (step 1). The credential server then further delegates the user's proxy credentials to a user proxy, which runs on the web portal (steps 2 and 3). From this point onward, the process continues just like before, with the web portal hosting the Grid software and relaying the communications to and from the user via HTTP/HTTPS. Similarly, firewalls (not shown in Fig. 3) can exist between the web portal and the remote Grid site; but these must allow general Grid traffic through [22].

III. Interactive GRID Security Issues

An Interactive Grid computing system [20] [23] is a Grid computing system supporting graphical, interactive sessions to remote users. Conceptually, it just consists of user nodes, a resource manager, and resources (see Fig. 4). The user submits job requests through a user node, and is given access to a remote resource for interactive use.

We have proposed an Interactive Grid architecture supported by Application Service Providers (ASPs) [24] to give a customer access to the desktop of a remote computer (see Fig. 5). To facilitate wide deployment, our solution works across firewalls at both the ASP site and the customer's site. The

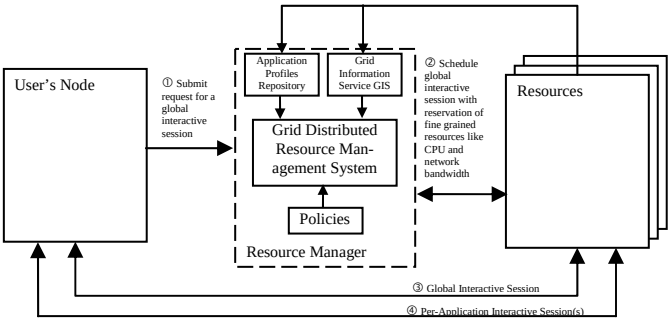

Fig. 4. Conceptual overview of an Interactive Grid Computing system

ASP's service is published as a web service in the Web Service Definition Language (WSDL). The initial request for service must be sent to this service access point accessible outside the ASP's firewall, and this communication should be compliant with the Open Grid Services Architecture [3].

The ASP site has several computing resources that are available for interactive use. Together, they constitute the resource pool. A firewall F1 protects the ASP's resources from denial-of-service and other attacks. Using the connection software on his node, the user connects to the Grid Service Access Point (GSAP) which has been published externally by the ASP as a web service. GSAP provides the functionality of the Gatekeeper in Figures 2 and 3. From GSAP, the request is forwarded to the Resource Manager, which now supports both interactive and batch jobs. It matches the resource requirements of the user to available resources by querying the Grid Information Service (see Fig. 4). The user might request an immediate allocation or an advance reservation. At the scheduled time, the Resource Manager instructs the software agent on the selected resource to start the remote display server RDS to connect to the communications server CS. The user also connects to the communications server using his remote display client RDC. CS facilitates communication between RDS and RDC by having an open port in the firewall F1. RDC displays the desktop of the remote host computer on the user's node.

In addition to the security mechanisms described in Section II for batch jobs, interactive GRID sessions have some additional security issues described below.

A. Fine Grain Access Control

Unlike a batch Grid job, a user is given explicit control of one or more allocated nodes in an Interactive Grid. This increases the potential for compromising system integrity by a malicious user. To provide added host system protection, we propose that the user be given only *controlled access*, consisting of a controlled shell, a controlled desktop, a controlled user account, and sometimes a controlled super user account.

Our *controlled shell*, called the Grid Interactive Shell (GISH) [20], only allows the user access to commands belonging to an allowed list of commands and runtime arguments. When GISH parses the command line typed by the user, a sequence of checks can be done. In addition to specifying an allowed list of commands and their options, the system administrator can also customize the directories and files to which the user is allowed access. This might be used to prevent a Grid user from looking at files containing sensitive information to which local users have access. Grid users will also have restrictions on programs compiled by them. In that case the application can be submitted to a trusted authority for certification as safe. Then the certified binary can be installed by the system administrator in a special directory, and added to the allowed list of commands. Alternately a virtual machine [25] with controlled access to disk and network can be provided, and the user can launch his binary within the virtual machine, without waiting for certification of his executables. In addition to being a shell for a *controlled user*, GISH can also be invoked as a shell for a *controlled super-user*. This privilege will of course be allowed only to a restricted set of users for administrative purposes. This might be needed when certain trusted Grid customers are given permission to assume super-user privilege for installation and updates for specific applications or services. In controlled super-user mode, GISH uses another set of policy files that govern the list of allowed commands and the access rights for directories and files.

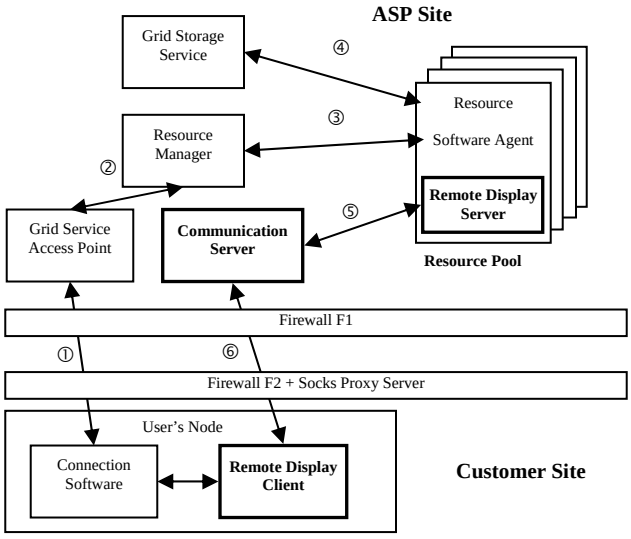

Fig. 5. Interactive Grid Architecture for Application Service Provider

We also provide a *controlled desktop* to the user. The *controlled desktop* has to be consistent with the *controlled shell* in terms of the policies enforced. The desktop's menus and icons can be customized by a file that is owned by root, and the user is not given permission to add or modify menu items or icons.

B. Interactive Session Management

An interactive session may consist of hierarchical sessions [23] - a global interactive session, and per application interactive sessions between the user's node and the resource (see Fig. 4). Each interactive session may have different security needs, like encryption and integrity of session communications. Interactive sessions also have relatively higher performance and response time constraints compared to batch jobs. Hence, security solutions deployed should be designed to minimize the performance overhead.

We propose a Communications Server to manage interactive session communications between the remote display client RDC on a user's node and the remote display server RDS on the ASP resource. The communications server functions as a proxy for the remote display protocol. For example, in our testbed implementation [24], we use VNC [26] as the remote display protocol and VncProxy [27] for the communications server. We assume a SOCKS proxy at the user's site. The remote display client connects to the communications server at the ASP's site through this SOCKS proxy server. This connection can be made secure, for example by establishing an SSL connection. The connection between the communications server and the remote display server can be similarly secured. The communications server can provide security functions for all the ASP resources, such as session encryption services. This reduces the amount of security processing required on each resource.

The communications server also assists with the more demanding needs for firewall traversal in interactive Grid sessions. The ASP typically protects its resources behind a firewall; the communication server has an open port in this firewall to facilitate remote display traffic and interactive session communications.

IV. New Security Scenarios and Vulnerabilities

We now propose a new approach to scoping Grid security issues in the context of our definitions of three generic scenarios: mutual trust, distrusted user and mutual distrust. Current Grid security architecture, described in section II, would belong in the first scenario of a mutually trusting environment.

A. Mutual Trust

The simplest form of Grid security is when both the user and the resource provider are mutually trusting. For example, the user and resource provider belong to an Intranet, which is composed of geographically distributed systems within the same company. In this environment, both the user and resource provider trust each other, but not potentially malicious third parties.

The Grid architecture and security is designed to protect against the actions of a third party, but may not always be effective. Mutual authentication ensures that the source of any request is really the user and that the user is issuing the requests to the correct resource provider. However, as in any network that is not physically protected, a third party might be able to eavesdrop on the communications. For many applications it may be desirable to use symmetric key bulk encryption for sensitive data being sent between the user and resource.

Additionally, the security of the mutual authentication process depends on the secrecy of the private keys belonging to the user, resource, and any certificate authorities. All of the hosts should check the certificate revocation lists to check for compromised keys and certificates. If a private key has been compromised, there will likely be a period of time before it is detected and reported. This provides an opportunity for malicious use of the Grid.

Some protection is provided by encrypting the user's private key with a password before it is stored. The user proxy's private key is unencrypted; however, the certificate is time-limited and will expire quickly without revocation. As a minimal precaution, all Grid activity could be logged to at least account for any damage done in any compromised scenario.

If a certificate authority (CA) has its private key compromised, the malicious third party can sign user and host certificates that will be accepted and trusted until the CA's certificate is revoked. This is mitigated by the fact that each CA is only trusted for certificates with distinguished names within its own virtual organization. Thus if one CA is compromised, other virtual organizations are not directly affected, except for any mappings they allow from the compromised virtual organization.

Any individual host could be compromised, potentially providing super-user access to a third party. On the user's site this is analogous to the private keys of the user and/or the user proxy being compromised or false requests being sent. If only the user proxy is compromised, then the time-limited nature of its key pair will limit the damage. If the user's private key is compromised, the threat is present until its certificate is revoked.

When a host on the resource's site is compromised, there is further potential for damage. First, any data or processes that are currently on the resource can be copied or manipulated. Many users will likely be active on the resource at any time and they will all be susceptible. The intruder may choose to sit silently to avoid detection and gather data for an extended period of time. He could also add new entries to the map file to use a resource in accordance with the Grid protocols when he would ordinarily be denied access. The intruder might use the stored credentials to impersonate the resource in dealings with other users that are not currently active. He could use the resource to launch some form of denial of service attack against another network. Alternatively the goal might be to take the resource out of service and deny access to the compromised resource to legitimate users.

When a web portal is involved, much of the security depends solely on password authentication. Brute force attacks or replay attacks might be used to gain access to the web portal or to a delegated user proxy by a third party. Beyond that, the credential server is likely protected by an internal firewall inside the web portal's site. Access to the web portal's host may provide additional access to the credential server. If the credential server becomes compromised, many users' credentials could be exposed and the users would not have remote access to the Grid until new credentials can be issued.

B. Distrusted User

The next Grid scenario of partial trust is when the user trusts the resource provider, but the resource provider does not necessarily trust the user. All of the security performed in the mutually trusting scenario is maintained, so the user has been authenticated and authorized. Additional security is added to control what the user can do on the resource.

Most resources already provide some level of user security through their local security policies. Usually the local user accounts are appropriately restricted by the operating system. This makes the operating system the first line of defense, using file permissions and any other built-in access control. A user will likely be unable to access the files or memory of other users on the same host. Even so, the Grid user will have the same access as regular local users, which may include sensitive information, internet and/or intranet network access, or local devices. The owner of the system may want to restrict Grid users further than other local users on the host.

A controlled shell like GISH described in Section III can provide the second line of defense against malicious users. The most basic vulnerability here is in the list of allowed programs and system calls. If this list is bypassed or modified, the user might get control of part of a resource he should not have access to. One such violation, even of a seemingly innocuous program, might open a door for further unauthorized access by the user.

The controlled shell might be designed with a fail-safe mechanism such that if some potential intrusion or modification is detected, the user would be locked out of running any further commands, instead of being allowed continued access to the system that might be partially compromised. After such an event occurs, significant effort may be required to re-establish a secure environment, since the user might have tampered with part of the system. It is also desirable to have an intrusion detection system (IDS) installed on the host to detect a failure of the controlled shell and limit how long a malicious user can have access. The IDS must try not to overwhelm the system administrators with false alarms since this will only desensitize them and allow real threats to be ignored. However this implies that the IDS must set some thresholds to determine what constitutes a potential intrusion and use them to filter out noise. One can expect malicious users to take advantage of this and try to evade detection.

C. Mutual Distrust

The last Grid scenario is when the user may also distrust the resource provider. This scenario can arise in many ways. The remote resources may be compromised or contain a loophole due to human errors or machine malfunction. This creates a window of vulnerability, during which the user's computation and data could be maliciously modified, or the confidentiality of the computation or data could be compromised. Second, the owners of the resource may themselves be malicious and intentionally use their access to the host to violate the protection the external user expects. For example, when a resource provider is another Grid user with spare computation time to share, he may take advantage of the trust the user placed in him.

Preserving the security and privacy of the user's data and computation is important. The data may be encrypted while it is stored. It may even be in the encrypted form after being transferred from the Grid storage server to the resource allocated to the user's computation. However, during the course of the computation, the data might be read from disk and decrypted in memory. If the host computer on which this computation is being done has been compromised, the unencrypted data might be read from memory and saved away by an intruder or malicious owner.

In a mutually distrusting environment, the user may desire some form of confirmation that his data has not been compromised as well as confirmation that it has been deleted with no copies kept after the job is completed. This would require some additional trusted level of hardware or software on the remote host. This extra level would need to be less susceptible to being compromised and access restricted even from its owner. The amount of distrust the user has of the remote site will dictate the level of extra security needed. In some cases, it may be acceptable to trust the Grid software, but not the other users. Or it may be acceptable to trust the operating system but not the Grid software. Finally, even the regular hardware may not be trusted and special security hardware may be necessary. Reference [28] suggests a mechanism for determining trust in a Grid environment. This is an area for future research to decide how much security is needed, how to provide it, and how to verify its correct operation.

V. Conclusions

In this paper we have discussed the security mechanisms in current Grid architectures, when a user connects directly to a Grid resource and when a web portal is added as an intermediate layer. The web portal allows the user to connect using only a web browser when their credentials are not available on the local computer. We introduce an architecture for Interactive Grids to expand from only using batch jobs, and discuss the new security mechanisms this requires. These are fine grain access control and interactive session management.

We then define three generic Grid security scenarios: mutual trust, partial trust (distrusted user), and mutual distrust. These decreasing levels of trust enable us to expose new vulnerabilities and show the increasing levels of security support required. In future work, we will attempt to propose solutions for these new security issues.

Acknowledgments

We would like to acknowledge Rob Atkins' work on installing the Globus Grid system at Princeton.

References

[1] I. Foster, C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", *Intl J. Supercomputer Applications*, 11(2):115-128, 1997

[2] I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International J. Supercomputer Applications*, 15(3), 2001

[3] I. Foster, C. Kesselman, J. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", Open Grid Service Infrastructure WG, *Global Grid Forum*, June 22, 2002, Available at http://www.globus.org/research/papers.html#OGSA

[4] *The Grid Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers; ISBN: 1558604758; 1st edition, November 1998.

[5] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, "A security architecture for computational Grids", *Proceedings of the 5th ACM conference on Computer and communications security*, November 1998.

[6] R. Butler, V. Welch, D. Engert, I. Foster, S. Tuecke, J. Volmer, C. Kesselman, "A national-scale authentication infrastructure", *Computer*, Volume: 33 Issue: 12 , Dec. 2000, Page(s): 60 -66.
[7] N. Nagaratnam, P. Janson, J. Dayka, A.Nadalin, F. Siebenlist. V. Welch, S.Tuecke, I. Foster, "Security Architecture for Open Grid Services". *GGF OGSA Security Work group*. http://www.ggf.org/ogsa-sec-wg/
[8] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowoski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke, "Security for Grid Services", *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing* (HPDC-12), Seattle, Washington, June 2003
[9] A.R. Butt, S. Adabala, N.H. Kapadia, R. Figueiredo, J.A.B. Fortes. "Fine-grain access control for securing shared resources in computational Grids", *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM* , 2002, Page(s): 206 -213.
[10] G. Buda, D. Choi, R.F. Graveman, C. Kubic, "Security standards for the global information Grid", Military Communications Conference, 2001. MILCOM 2001. *Communications for Network-Centric Operations: Creating the Information Force*. IEEE, Volume: 1, 2001, Page(s): 617 -621 vol.1.
[11] M. Humphrey, M. Thompson, "Security Implications of Typical Grid Computing Usage Scenarios", *Security Working Group GRIP forum draft*, October 2000.
[12] P. Ning , S. Jajodia , X.S. Wang, "Abstraction-based intrusion detection in distributed environments", *ACM Transactions on Information and System Security* (TISSEC) Volume 4 Issue 4, November 2001.
[13] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke, "A Community Authorization Service for Group Collaboration", *Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*, 2002.
[14] C.E. Phillips, T.C. Ting, S.A. Demurjian, "Mobile and Cooperative Systems: Information sharing and security in dynamic coalitions", *Seventh ACM Symposium on Access Control Models and Technologies*, June 2002.
[15] S. Tuecke et. al., "Internet X.509 Public Key Infrastructure Proxy Certificate Profile", *IETF PKIX Working Group Draft*
[16] R. Atkins, J. Dwoskin, "Princeton Globus Installation Homepage." http://palms.ee.princeton.edu/globus/
[17] The Globus Project, http://www.globus.org/.
[18] T.J. Hacker, B.D. Athley, "A methodology for account management in grid computing environments", *Second International Conference on Grid Computing*, November 2001
[19] An Accounting System for the Datagrid Project version 3.0. http://server11.infn.it/workload-grid/docs/DataGrid-01-TED-0115-3_0.pdf.
[20] V. Talwar, S. Basu, R. Kumar, "An Environment for Interactive Grids", *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing* (HPDC-12), Seattle, Washington, June 2003
[21] J. Novotny, S. Tuecke, V. Welch. "An online credential repository for the Grid: MyProxy", *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, 2001, Page(s): 104 -111.
[22] V. Welch, "Globus Toolkit Firewall Requirements", http://www-fp.globus.org/security/v2.0/firewalls.html, 2003.
[23] R.Kumar, V.Talwar, S.Basu, "A Resource Management Framework for Interactive Grids", *First International Workshop on Middleware for Grid Computing*, Middleware 2003, Rio de Janeiro, Brazil, June 2003
[24] S. Basu, V. Talwar, B. Agarwalla, R. Kumar, "Interactive Grid Architecture for Application Service Providers", *First International Conference on Web Services*, Las Vegas June 2003
[25] E. Bugnion, S. Devine, K. Govil, and M. Rosenblum. "Disco: Running commodity operating systems on scalable multiprocessors." *ACM Transactions on Computer Systems*, 15(4):412--447, 1997
[26] T. Richardson, Q. Stafford-Fraser, K. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33-38, Jan/Feb 1998
[27] VncProxy, http://www.wilson.co.uk/Software/vnc/proxy/VncProxy.htm
[28] F. Azzedin, M. Maheswaran, "Towards trust-aware resource management in Grid computing systems", *Cluster Computing and the Grid 2nd IEEE/ACM International Symposium* CCGRID2002, 2002, Page(s): 452 -457.

APPENDIX A: AUTHENTICATION USING CERTIFICATES

Grid Security Infrastructure primarily uses an authentication algorithm defined by the Secure Sockets Layer Version 3 (SSLv3) protocol, illustrated in Fig. A-1.

Suppose Alice wants to authenticate herself to Bob, a remote host. First, Alice presents her certificate to Bob. Bob will then provide Alice with a unique piece of random data called a nonce. Alice signs (encrypts) the nonce with her private key and sends the result to Bob. Bob gets Alice's public key from her certificate and uses this to decrypt the data sent by Alice. If the decrypted data matches the nonce Bob originally sent, then he knows it must have been signed using the corresponding private key.

The question remaining is whether or not the public key in Alice's certificate really belongs to her. This is where the CA comes in. The CA signed Alice's certificate with its private key. Bob maintains a copy of the CA's public key for each CA he trusts. Therefore Bob uses the CA's public key to verify the signature on Alice's certificate. If it matches, then the certificate Alice presented must have been unchanged from when the CA signed. Now Bob can trust all of the information in Alice's certificate since it was verified by the CA.

Also, a chain of certificates can be formed where each member in the chain signs the certificate of the member below it. The trusted CA sits at the top of the chain. Alice can also delegate her authority to another entity (e.g., her user proxy) by signing its certificate with her private key, forming a certificate chain.

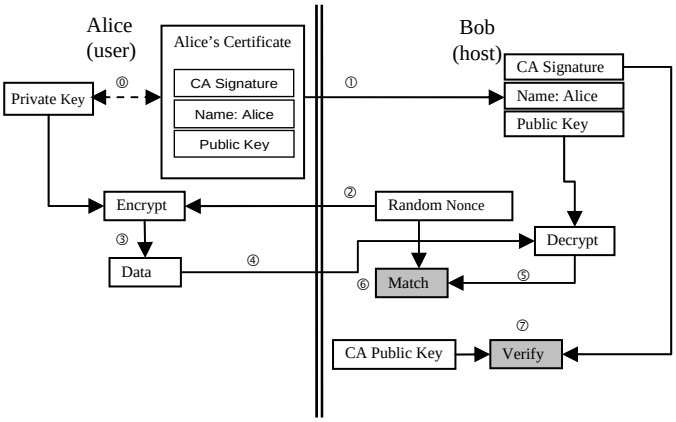

Fig. A-1. Authentication process