An Online Credential Repository for the Grid: MyProxy

Jason Novotny
Lawrence Berkeley Laboratory*
JDNovotny@lbl.gov

Steven Tuecke
Mathematics and Computer Science Division
Argonne National Laboratory
tuecke@mcs.anl.gov

Von Welch
Distributed Systems Laboratory†
University of Chicago and
Argonne National Laboratory
welch@mcs.anl.gov

Abstract

Grid Portals, based on standard Web technologies, are increasingly used to provide user interfaces for Computational and Data Grids. However, such Grid Portals do not integrate cleanly with existing Grid security systems such as the Grid Security Infrastructure (GSI), due to lack of delegation capabilities in Web security mechanisms. We solve this problem using an online credentials repository system, called MyProxy. MyProxy allows Grid Portals to use the GSI to interact with Grid resources in a standard, secure manner. We examine the requirements of Grid Portals, give an overview of the GSI, and demonstrate how MyProxy enables them to function together. The architecture and security of the MyProxy system are described in detail.

1 Introduction

Grid technologies [9, 8] are increasingly becoming the platform of choice for developing and deploying distributed computation and data intensive applications across large virtual organizations, such as the NASA Information Power Grid[13] and the NSF PACI Grids[2, 3]. Grid Portals are a common approach to providing user interfaces to such Grid applications. By combining a web server and Grid-enabled software, a Grid Portal allows the use of a standard Web browser as a simple graphical client for Grid applications[14].

Unfortunately, standard Web security protocols, employed between a Web client and server, do not support the needs of Grid Portals. Most Grid Portals require that the user delegate to the server the right for that server to act on the user's behalf, in order to initiate and monitor operations for that user on Grid resources. Such Grid resources are generally protected by the Grid Security Infrastructure (GSI)[5], a component of the Globus Toolkit[7] that has become the de-facto standard for Grid security. While GSI supports such delegation, the standard Web security protocols do not. This leads to an incompatibility between Web and Grid security, which must be overcome in order to enable the smooth operation of Grid Portals.

This paper describes an online credentials repository, called MyProxy, designed and developed by the authors in order to bridge this incompatibility between Web and Grid security, thus enabling Grid Portals to use GSI-protected resources in a secure, scalable manner. Section 2 describes the GSI and defines security terms and concepts used throughout the paper. Section 3 explains the issues that required the development of a credentials repository to enable Grid Portals. The MyProxy system is described in detail in Section 4, along with a description of how it works to enable a Grid Portal. Section 5 gives a detailed discussion of security considerations with the MyProxy system, and Section 6 details future research and development directions.

2 Overview of Grid Security Infrastructure

The Grid Security Infrastructure (GSI) is a set of protocols, libraries, and tools that allow users and applications to securely access Grid resources. In this section we describe GSI and define the security terms and concepts discussed in later sections of the paper.

*National Laboratory for Applied Network Research and National Center for Supercomputing Applications at time of work

†National Center for Supercomputing Applications at time of work

2.1 Public Key Infrastructure

The GSI system is based on a Public Key Infrastructure (PKI). In a PKI, all entities (users and resources) are identified by a globally unique name known as a Distinguished Name (DN). Authentication with the GSI is a matter of proving that a user or resource is the entity identified by a DN. Resources then typically have local configuration for mapping the DN to a local identity (e.g. Unix hosts have a file containing DN and username pairs).

In order for entities to prove their identity, they possess a set of Grid credentials consisting of a certificate and a cryptographic key known as the private key. The certificate, in simple terms, is a binding of the entity's DN to their private key. This binding is done by means of a digital signature from a trusted party known as a Certificate Authority (CA). This allows an entity to authenticate (using their credentials) by a process that involves presenting their certificate and proving possession of their private key.

An important thing to note about credentials in a PKI environment is that an entity must have sole possession of its private key to maintain the integrity of the system. If another party were to gain possession of the private key, the second party would be able to impersonate the owner at will and without restraint until the theft was discovered and the certificate revoked by the CA.

To limit the danger of an entity's private key being stolen, two things are usually done. First it is typically protected from unauthorized access in some manner. This is generally done by storing the key in a file with restricted access, storing it in an encrypted file with a decryption pass phrase known only to the owner, or storing it on a hardware token (e.g. a smart card) that requires a pass code. The hardware token provides the best security but is in very limited use due to the lack of deployment of hardware support for using the tokens.

The second method of protecting a private key is by giving a credential a limited lifetime after which it is no longer valid. This requires entities to regularly acquire a new set of credentials, which protects the private key by limiting its exposure. Typically this lifetime is on the order of years, with the exact length of time determined by the policy of the CA that issued the certificate.

2.2 Secure Socket Layer

GSI uses Secure Socket Layer (SSL)[11] to implement authentication (providing identity to another entity), message integrity (making sure no one has modified a message between two entities) and message privacy (making sure no one has can read a message between two entities). SSL is a standard software tool used in web browsers, web servers, and other applications. It uses PKI credentials for authentication and is used in GSI without modification.

2.3 Proxy Credentials

In the Grid environment, a user may need to authenticate themselves multiple times in a relatively short period of time, for example if multiple resources are being coordinated. Requiring the user to repeatedly type their pass phrase for the multiple authentications is clearly undesirable from a convenience standpoint. It is also undesirable from a security standpoint, as each time their private key is decrypted, it is another opportunity for it to be compromised. An alternative is for the software to only prompt for the user's pass phrase once, but retain either the pass phrase or the unencrypted private key for multiple usages. However, while this is much more convenient for the user, this is even more undesirable from the security standpoint, as it exposes the private key to attack for a long period of time.

GSI solves this problem with proxy credentials. A proxy credential is a short-term credential that is created by a user, which can be used in place of the long-term credential to authenticate that user. The proxy credential has its own private key and certificate, and is signed using the user's long-term credential. The proxy certificate, in effect, is a short-term binding of the user's DN to an alternate private key. Proxy credentials are stored unencrypted on the local file system, protected only by file system permissions, and so can be used by the user repeatedly without inconvenience. Because a proxy credential is more vulnerable to compromise, they are typically given much shorter lifetimes than the user's long-term credentials; usually on the order of hours or days.

2.4 Delegation

It is often important in distributed applications for a user's application to be able to act, unattended, on the user's behalf on the Grid. An example of this is a user's job that needs to be able to authenticate as the user to mass storage system to store the result of a long computation. The GSI solves this problem by allowing the user to delegate a proxy credential to processes on remote hosts.

Delegation is very similar to proxy credential creation as described in Section 2.3 in that an existing set of credentials is used to create a new set of proxy credentials that is identical in function. The difference is that the creation occurs over a GSI-authenticated connection, with the result being the remote process acquiring proxy credentials for the user.

It is also worth noting that delegation can be chained. In other words one can delegate credentials to host A and then the process on host A can delegate credentials to host B and so forth.

2.5 Typical GSI Usage

A typical session with GSI would involve the user using their pass phrase and a GSI tool called grid-proxy-init to create a proxy credential from their long-term credential. The user could then use a GSI-enabled application, such as the Globus Toolkit's GRAM [6] or Secure Shell (SSH), to connect to a remote host. The application would use the GSI library and the user's proxy credential to authenticate to the remote host and (assuming the user requested so) delegate a proxy credential to the remote host. The process running on the remote host could then further authenticate with GSI to other hosts and potentially delegate further proxy credentials.

3 The Need for a Credential Repository

In this section we explain the need for the MyProxy repository. We first examine the requirements of Grid portals, then the constraints of the existing software that prohibited meeting these requirements, and we then lay out the design goals for the MyProxy system.

3.1 Grid Portal Requirements

Discussions with the portal development community revealed a number of requirements that impact security. These included:

- Users must be able to use any standard web browser to access the Grid portals.
- Users must be able to use a web browser from locations where their Grid credentials would not normally be available to them.
- Users must be able to do anything through a Grid portal that their credentials would entitle them to do.

For example, a user should be able to access the Grid using a web browser at an airport kiosk in the same manner as they could from a web browser installed on a system on their desktop in their office.

3.2 Existing Software Constraints

Existing software has two constraints when it comes to meeting Grid Portal requirements. First, since Grid credentials are typically stored as files on a file system and the private key must be kept private, a user must have secure access to the file system to use their credentials. This typically means that the user must be sitting at the machine, or logged in a secure manner (e.g. an encrypted Secure Shell session). This often results in a user being unable to access their credentials when away from their primary system. While smart cards and other hardware tokens (which can store credentials on a portable medium) would be a potential solution to this problem, support for these devices has not been widely deployed among the broad range of users of Computational Grids.

The second constraint is that not all applications that are desired for Grid use are GSI-enabled, and hence lack the ability to do proxy credential delegation. An example of this is the use of web browsers as clients with web-based portals. Web browsers can use Grid credentials for authentication, but lack the GSI's ability to do credential delegation as described in Section 2.4. This means that although the user can authenticate to the portal, they cannot delegate authority to the portal to act on their behalf, meaning the portal is severely restricted in usefulness.

Due to the above constraints, Grid portals were previously forced to give the portal permanent special privileges to act on a user's behalf on the Grid. This was typically done by giving the portal possession of the user's long-term credentials. However, having every portal that a user wants to use have a copy of the user's credentials is highly undesirable. The foremost reason is that the more copies of a user's credentials that exist, the greater the exposure of the credentials and the greater the risk of the credentials being stolen. In addition, requiring the portal to securely manage all its user's long-term credentials greatly increases the security requirements on the operation of portal itself.

3.3 Goals for the MyProxy System

The combination of the portal requirements and limitations brought about the need for a system to meet a number of goals:

- It should allow users to access their credentials from anywhere on the Grid, even if they are on a system without Grid software and without secure access to their long-term credentials.
- It should allow them to delegate credentials to resources to which they normally would not be able to, since the applications involved do not support the GSI delegation mechanism (e.g. from a web browser to a portal).
- It should remove, as much as possible, any credentials from the portal except when they are actually needed, in order to lower the risk of compromise if the portal is compromised.
- It should be scalable. Multiple portals should be able to use a single system in the case of a domain having more than one portal, and a portal should be able to use

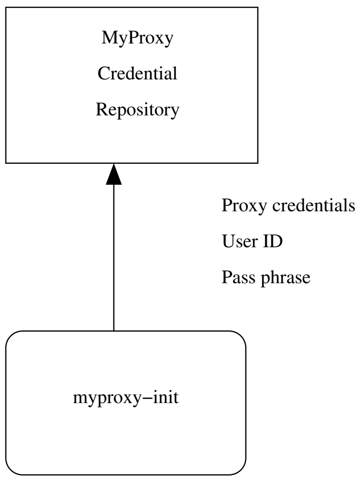

Figure 1. MyProxy-init process

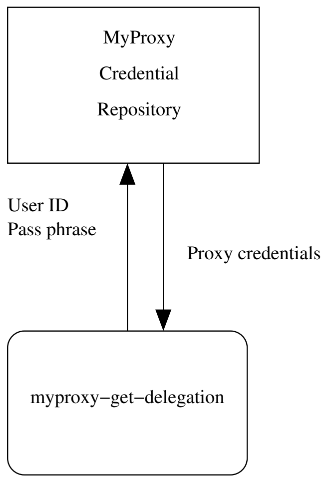

Figure 2. MyProxy retrieval process

multiple systems in the case of a portal that supports users from multiple domains.

- It should give the user as much control of their credentials and proxy credentials as possible. Portals should only be able to get a user's credentials if allowed to do so by a user.

4 The MyProxy Credential Repository System

In this section we describe the MyProxy Online Credential Repository and client tools that we designed and developed in order to meet the goals as layed out in Section 3.3. We first describe the two basic steps of using the repository: delegation of a proxy credential to the repository, and retrieving the credential from the repository. We then show how the MyProxy repository system is used with a Grid portal. Finally we describe the various implementations of the MyProxy repository and tools that exist today.

4.1 Delegation to the Repository

The MyProxy credential repository system consists of a repository server and a set of client tools that can be used to delegate to and retrieve credentials from the repository. Normally, a user would start by using the myproxy-init client program along with their permanent credentials to contact the repository and delegate a set of proxy credentials to the server along with authentication information and retrieval restrictions (See Figure 1).

The authentication information in this process consists of a user identity and a pass phrase to be used to authenticate any later retrieval operations. Both the user identity and pass phrase are chosen by the user, but can be tested by the repository to make sure they meet any local policy (e.g. the pass phrase must be a certain length, survive dictionary checks, etc.). The user identity is also typically different from the user's Distinguished Name (DN) in their Grid credentials, as it is actually hand-typed by the user at later times. Because of this it is desirable for the user identity to be more memorable and concise than a typical DN.

The retrieval restrictions are currently limited to a maximum lifetime for proxy credentials that the repository may delegate on the user's behalf. See Section 6.5 for description of our plans to expand these restrictions.

The credentials delegated to the repository normally have a lifetime of a week. The user can change this to any length of time desired. The user can also, at any point, use the myproxy-destroy client program to destroy any credentials they previously delegated to the repository.

4.2 Retrieval of Credential from Repository

At a later time, the user, or a service acting on behalf of the user, uses the myproxy-get-delegation program to contact the server and request a delegation of the user's credentials (See Figure 2). The user must supply the same user identity and pass phrase that they provided when initially delegating the credentials to the server. After verifying this information and checking any restrictions that the user presented with the delegation (see Section 6.5), the repository will in turn delegate a proxy credential back to the user or service. This delegated proxy from the repository may then be used as any other proxy credential generated by the user to initiate actions on the user's behalf on the Grid.

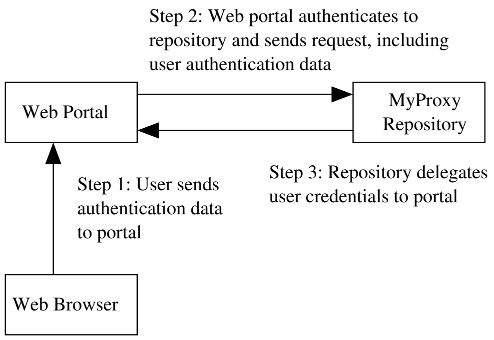

Figure 3. User connecting to portal and portal interaction with MyProxy repository

4.3 Using the MyProxy Repository with a Grid Portal

The first step to using MyProxy system with a portal is to delegate a proxy credential to the repository as shown in Figure 1 above. Then, at a potentially different time and place, the user would connect to a Grid portal (using a web browser) and supply the authentication information given earlier to the repository through a web form or similar interface (see Figure 3, step 1). The user might also specify a MyProxy repository for the portal to use, assuming it's configured to use more than one.

The portal would then connect to the MyProxy repository using the myproxy-get-delegation program, authenticates itself using it's own Grid credentials, presents the authentication information (user identity and pass phrase) as provided by the user, and requests a proxy credential for the user (Figure 3, step 2).

The repository then, after verifying all the presented information, would delegate a proxy credential for the user back to the portal (Figure 3, step 3). The portal then can securely access the Grid using standard Grid applications as the user normally would. At this point the user would direct the portal through the existing connection with the web browser.

The operation of logging out of the portal deletes the user's delegated credential on the portal. If a user forgets to log off, than the credential will expire at the lifetime specified when requested from the MyProxy service. This lifetime is normally on the order of a few hours.

This process could then be repeated as many times as the user desires until the credentials held by the MyProxy repository expire, at which point the user would need to rerun the myproxy-init program from a location where their permanent credentials were available and delegate a new set of credentials to the repository. The maximum lifetime of credentials delegated to the repository is set by policy on the repository server, but defaults to one week.

4.4 Implementations

There is currently available a C implementation of the MyProxy server and client programs (ftp://ftp.ncsa.uiuc.edu/aces/myproxy/), along with Java implementations of the MyProxy clients included in the Java CoG kit (http://www.globus.org/cog/). The Java MyProxy implementation includes a client API for accessing the MyProxy server as well as graphical tools useful for end users to delegate credentials to the MyProxy server.

5 Discussion of Security Considerations

In this section we discuss some of the security decisions inherent in the design of the MyProxy system, as well as policy decisions that are made by MyProxy system and Grid portal installers. We also discuss some the risks particular to the MyProxy system, and the steps that we have or are taking to minimize these risks.

5.1 Repository Issues

The obvious risk of the MyProxy system is the collection of delegated user credentials that are held by the MyProxy repository. If the host on which the repository is compromised, an attacker could potentially gain access to these credentials. Because of this, the repository should only be run on a tightly secured host (e.g. comparable to a Kerberos Domain Controller). Additionally, to minimize this risk, the repository encrypts the credentials that it holds with the pass phrase provided by the user. Because of this, even if the repository host is compromised, an intruder would still need to decrypt the keys individually or wait until a portal connects and provides a pass phrase from a user. In either case, the required delay allows credentials to expire or for the intrusion to be detected.

The MyProxy repository authenticates all incoming connections, restricting service to authorized clients. A list of authorized clients is defined by two access control lists, one for clients allowed to delegate to the repository (typically users), and a second for clients allowed to request delegations from the repository (typically portals). The latter is particularly important, as it prevents unauthorized clients from retrieving a user proxy from the repository, even if such clients are able to gain access to the user's MyProxy authentication information. Replay attacks by such a client via a valid portal could be prevented by replacing the current MyProxy pass phrase scheme with a one-time password system[12].

MyProxy clients also require mutual authentication of the repository through the use of Grid credentials held by the server. This prevents an attacker from impersonating the repository in order to steal credentials or authentication information.

Another risk is the compromise of a portal that is privileged to contact the MyProxy server and request delegations. This risk is minimized by the fact the MyProxy server requires the user authentication information in addition to the authentication of the portal. This requires that the intruder wait for the user to connect and provide this information, which allows time for intrusion to be detected or credentials to expire.

Since sensitive information is transferred between the MyProxy client programs and the server, all data passing to and from the server is encrypted GSI's message confidentiality mechanisms (i.e. using SSL). It should be noted that this is currently needed because the MyProxy server uses a persistent pass phrase for authenticating the user, which would allow an eavesdropper to perform replay attacks. The current pass phrase scheme could be replaced with a one-time password system[12] eliminating this need for message confidentiality.

5.2 Issues specific to the Grid Portal

The portal web server must currently be configured to only allow HTTP connections secured with SSL encryption (HTTPS), since transmitting the name and pass phrase over unencrypted HTTP would allow any intruder to snoop the pass phrase over the network. While the MyProxy repository's client authorization prevents direct (mis-)use of the captured pass phrase, the compromised pass phrase could be used in a replay attack against the portal. Using a one-time password would lift this HTTPS restriction.

It's also worthy to note that the credentials maintained by the portal, which it uses to authenticate to the MyProxy server, are unencrypted. This means the portal does not require any additional administration to retrieve credentials on behalf of users. For additional security, a portal could maintain encrypted credentials that could be decrypted by the web server administrator upon startup and made valid for a chosen lifetime. In practice, since the portal is considered to be a production service, requiring an administrator to constantly decrypt the portal key can be a significant management burden.

It is also the portal's responsibility to not only maintain the user's credentials while in use, but to map the credentials to the user's web session. This requires session tracking between clients and servers by transmitting identifying data between the client and server. Because HTTP is a stateless protocol, this is often accomplished with cookies or rewriting URLs to include additional data and establish a session. By maintaining state between the client and browser, the portal can uniquely identify a user associated with the user's credentials. When a user makes a request to perform a remote task, such as file transfer or job submission, the portal can use the identifying information to determine the credential to be used in authenticating to the selected Grid resource.

Current implementations of portal software are based on CGI and Java Servlet technology. The GridPort project at SDSC uses Perl CGI scripts[14] that simply wrap the myproxy-get-delegation program with the appropriate user identity and pass phrase passed on from the HTTPS request header. The Grid Portal Development Kit[1] uses the Java Servlet API to manage sessions and uses the Java CoG toolkit MyProxy API to access the MyProxy server.

6 Future directions

This section describes some of the ongoing research and development with the MyProxy system and in the GSI system as it pertains to MyProxy.

6.1 Managing permanent user credentials

Traditional Public Key Infrastructure (PKI) security systems, like the GSI, store user identity credentials on disk, encrypted with a password. Our experience with GSI's use of this approach has shown that most users dislike managing their own identity credentials in this manner. Doing so is error prone (and thus less secure) and sometimes results in the credentials not being available from where they are needed by the user.

We plan to address this problem by developing the current MyProxy system into a more general online credential repository system capable of managing long-term Grid credentials on the user's behalf. When the user needs to use the credential, a client is used to authenticate with the repository using local site security (e.g. Kerberos), a name and pass phrase, or one-time-password, etc. In response, the repository will use the user's long-term credential to delegate a proxy credential back to the client. The IETF SACRED working group[4] is addressing a similar problem and may provide valuable leverage in this area.

6.2 Managing multiple user credentials

As the number of organizations and CAs grow it is inevitable that users will end up with multiple credentials that be may required concurrently, sometimes for the same computational job. The complexity of having to determine and select the correct credentials to use for a particular task is likely to be a large burden for these users. We plan to investigate having the credential repository act as an electronic wallet – a storage mechanism for all of a user's credentials. This wallet would be able, when given information about the task a user wishes to undertake, to correctly select credentials for the task, embed the minimum needed rights in those credentials, and then return the credentials to the user.

6.3 Alternate authentication mechanisms

We plan to investigate replacing the current user identity and pass phrase authentication mechanism with more advanced schemes (e.g. one-time pass phrases) and existing local site security mechanisms (e.g. Kerberos).

6.4 Standardize client-server protocol

The current MyProxy client-server protocol was quickly designed as a prototype. We plan to investigate using more standard protocols. One options would be HTTP for compatibility with standard web-oriented libraries. A standard protocol for accessing on-line repositories produced by the IETF SACRED working group[4] would also be an option.

6.5 Restricted Proxy Credentials

Currently the only available restriction that can be placed on delegations by the repository is the lifetime of credentials. However there is standard under development in the Global Grid Forum and IETF for placing fine-grain restrictions on delegated credentials[15, 16]. This would allow users to explicitly place limitations on the credentials they delegate to the MyProxy server, so that even if the MyProxy server itself were compromised or the credentials themselves were somehow stolen, the damage that could be done with them would be significantly limited.

6.6 Support for Condor-G

It is not uncommon for computational jobs to run for a period of time that exceed the lifetime of the proxy credential they receive on startup. The Condor-G system[10] provides a support for this by emailing a user when they need to refresh their credentials. However this can be inconvenient for the user. We plan to investigate mechanisms to enable MyProxy to securely support long-running applications by being able to supply them with fresh credentials when needed.

7 Summary

This paper describes an online credentials repository, called MyProxy, designed and developed by the authors in order to enable Grid Portals to access Grid Security Infrastructure protected resources in a secure, scalable manner. We explain the issues that required the development of an online credentials repository to enable Grid portals, describe the MyProxy system, and demonstrate how it works to enable a Grid Portal. Finally we give a detailed discussion of security considerations with the MyProxy system, and a description of future research and development directions.

The MyProxy repository is currently deployed in several production portals including the National Computational Science Alliance, NPACI, and the NASA Information Power Grid. This experience has shown that online credential repositories can play an important role in enabling easier use of Grids, such as with Grid Portals. Further, we believe that with the added functionality of restricted delegations, management of permanent user credentials, and automatic selection of credentials from multiple credentials, these online credential repositories could serve to greatly improve the user's interface to Grid security and prove useful for additional application level services beyond portals.

8 Acknowledgements

We would like to thank the following people for contributing ideas, requirements, testing, and/or feedback to MyProxy: Randy Butler, Doug Engert, Jarek Gawor, Steve Mock, Benjamin Temko, Mary Thomas, Keith Thompson, and Gregor von Laszewski. Apologies to anyone we missed.

MyProxy was developed under funding from the National Center for Supercomputing Applications and the National Laboratory for Applied Network Research. This work was also supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38; by the Defense Advanced Research Projects Agency under contract N66001-96-C-8523; by the National Science Foundation; and by the NASA Information Power Grid project.

References

[1] The Grid Portal Development Kit. `http://dast.nlanr.net/Features/GridPortal`.

[2] National Computational Science Alliance home page. `http://alliance.ncsa.edu/`.

[3] National Partnership for Advanced Computational Infrastructure home page. `http://www.npaci.edu/`.

[4] A. Arsenault and S. Farrell. Securely available credentials – requirements, IETF Internet Draft draft-ietf-sacred-reqs-01.txt, Feb. 2001.

[5] R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer, and V. Welch. Design and deployment of a national-scale authentication infrastructure. *IEEE Computer*, 33(12):60–66, 2000.

[6] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Tuecke. A resource management architecture for metacomputing systems. In *IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*.

[7] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *Intl J. Supercomputer Applications*, 11(2):115–128, 1997.

[8] I. Foster and C. Kesselman, editors. *The Grid: Blueprint for a Future Computing Infrastructure*. 1999.

[9] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. In *To be published in Intl. J. Supercomputer Applications, 2001*.

[10] J. Frey, T. Tannenbaum, I. Foster, M. Livny, and S. Tuecke. Condor-g: A computation management agent for multi-institutional grids. In *Proc. of the Tenth IEEE International Symposium on High Performance Distributed Computing*.

[11] A. Frier, P. Karlton, and P. Kocher. The SSL 3.0 protocol. Technical report, Netscape Communications Corp., Nov. 1996.

[12] N. Haller, C. Metz, P. Nesser, and M. Straw. A One-Time Password system, IETF RFC 2289.

[13] W. E. Johnston, D. Gannon, and B. Nitzberg. Grids as production computing environments: The engineering aspects of nasa's information power grid. In *Proc. of the Eighth IEEE International Symposium on High Performance Distributed Computing*.

[14] M. Thomas, S. Mock, and J. Boisseau. Development of web toolkits for computational science portals: The NPACI HotPage. In *Proc. of the Ninth IEEE International Symposium on High Performance Distributed Computing*.

[15] M. Thompson, D. Engert, and S. Tuecke. Internet X.509 public key infrastructure restricted impersonation certificate profile, Global Grid Forum draft-ggf-x509-res-delegation-01.txt, Feb. 2001.

[16] S. Tuecke, D. Engert, and M. Thompson. Internet X.509 public key infrastructure proxy certificate profile, Internet Draft draft-ietf-pkix-impersonation-00.txt, Feb. 2001.