GEON: Standards-based secure invocation of Arcweb services.

Ashraf Memon, Chaitan Baru, Ilya Zaslavsky, Steve Mock
(amemon|baru|zaslavsk|mock)@sdsc.edu

Abstract

Within the Geosciences Network (GEON) project, we implemented a secure mechanism of data interchange across distributed spatial databases and services, a common requirement in many geographic applications. The paper explores OGSI-based grid services that implement Grid Security Invocations (GSI), and describes an application of this model to secure invocation of ArcWeb services.

Currently, ArcWeb services use single sign-on security model, in which, on initial authentication, the user is assigned a time-limited numeric token that is passed along with the request message and used to unlock an ArcWeb service. We implemented a GSI authentication wrapper on top of ESRI's token-based authentication mechanism. To invoke an ArcWeb service on the secure grid, the user first requests a time-limited proxy certificate by providing a user name and pass phrase to a certificate authority. This certificate, which is transmitted inside SOAP headers of all service requests, is verified each time before a service is invoked.

Introduction

Web services represent an increasingly popular standard for exposing data and functionality on the Internet. ESRI's Arcweb services [1] are a pool of web services that provide mechanism to access data and GIS functions on demand over the World Wide Web. Benefits of using the web services approach for geographic applications include:

- Seamless standards-based access to terabytes of data without having to manage them by yourself.
- Easy on-demand standards-based invocation of GIS capabilities without the need to install software on the desktop,
- Added confidence that data and services retrieved or invoked remotely, are up to date, developed by a reputable source and passed expert review.
- For developers: ability to incorporate remote services and data into custom applications, which leads to considerable development savings

Despite rapid proliferation of web services-based approaches, their implementation in environments, which require secure access to data and services, remains a challenge. Part of the reason is lack of standards for handling security in the current web services model. In this paper, we review available web services security standards, and describe the security model followed by ESRI's Arcweb Services. We then propose enhancements to the Arcweb security model based on GSI authentication, and describes GSI authentication for web services as implemented in the NSF-funded Geosciences Network (GEON) project. The paper concludes with a summary and outline of future work.

1. Web Service Security.

Web service security has several components [2] all of which must be addressed by a comprehensive solution. They include:

1. Authorization: the process of assigning a user to an account and verifying this account when the user accesses it.
2. Authentication: making sure the user is actually the one who she/he is claiming to be.
3. Non-repudiation: keeping a permanent non-modifiable record of every action the user takes.
4. Integrity: ensuring that the message exchanged between the systems is not tampered with.

5. Confidentiality: ensuring that the information can only be accessed by authorized users.

Several protocols have emerged to address some of these areas of web service security. Some of them describe code signing for authentication of request-response messages [3]: until the code is verified the user is not authenticated and communication does not proceed. Others address the issue of securing the network channel on which the communication occurs. Another group of protocols focus on securing the content of the message rather than the connection, providing for this content to be opened only by an authorized person with the right to decrypt the message. Following are the popular protocols/specifications that are related to web service security:

- **SAML (Security Assertion Markup Language)**: This protocol is used in authentication and authorization of web service requests and responses that allows a single sign on.
- **XML Digital Signatures**: This specification addresses the issue of digitally signing the message to ensure that the message maintains its integrity.
- **XML Encryption**: This specification describes encryption of XML messages depending on an encryption scheme agreed-upon between parties.
- **WS-Security**: WS-security addresses most of the issues described in the previous paragraph and defines solution by making use of the SOAP headers.

Within the GEON project, we have explored yet another security mechanism for web services based on Grid Security Infrastructure (GSI) [5]. GEON develops grid services-based infrastructure for sharing geoscience datasets, hence we focus on security mechanisms developed within the grid community. Grid Security Infrastructure (GSI) is based on proven standards such as public key encryption, X.509 certificates [4], and the Secure Sockets Layer (SSL). To access a GSI-secured system, each user or a group of users must be assigned a public and a private key. The public key, which is generated and distributed by an entity known as Certificate Authority (CA), can be made available to everyone while the private key must be kept secret [4]. GSI authentication uses an extension of X.509 certificates to provide the single sign-on capability. Each user certificate known as proxy certificate contains user's credentials. The benefits of using GSI include:

1. Security in message communication between various actors involved in the information interchange
2. Ability to support security across organizational boundaries
3. Ability to provide single sing-on system.

Generally, a certificate in this case includes four attributes, which define identity of the invoker, public certificate of the invoker, identity of the CA, and digital signature of the CA. This certificate identifies each user or a group of users specifically along with the issuing CA [6].

Web services use Simple Object Access Protocol (SOAP) for sending and receiving messages between service invoker and service provider. A SOAP-formatted message has two parts, a header and a body. The SOAP body mainly carries request parameters and results, while the header is used to carry meta information which could include security information, routing information etc. Below, we use the GSI mechanisms to extend the token based security implemented in Arcweb services, by adding GSI authentication to the headers ot SOAP messages used to communicate with the services. The idea is to implement this GSI authentication layer as a wrapper on top of the current ArcWeb security and provide a generic secure authentication mechanism for invoking web services.

GSI Authentication for Arcweb Services

1. Existing security mechanism in Arcweb Services:

To access ArcWeb services from the client, the invoker is required to have an ArcWeb account. ESRI offers a wide range of account options for its ArcWeb services which provide access to different sets of services and depend on number of allowed service invocations.. Once the account is created, the user receives username and password that are used to invoke the web services. Figure 1 shows the currently implemented mechanism.

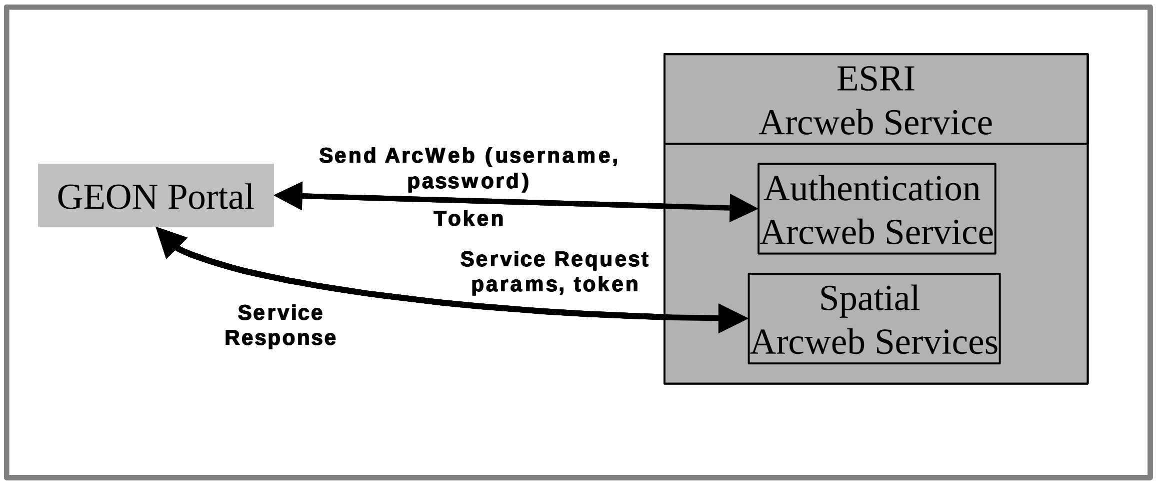
ESRI
Arcweb Service
Send ArcWeb (username, password)
GEON Portal
Token
Authentication Arcweb Service
Service Request params, token
Spatial Arcweb Services
Service Response

Figure 1. Existing Model of security in Arcweb Services

Once a user is identified as the authorized user based on the submitted username and password (over secure HTTPS connection), he/she receives a numeric token from ESRI which is used to access Arcweb Services. The assigned token must be sent along with each request submitted to ArcWeb servers. This security token has a default timeout. The benefits of this authentication scheme include

- Minimizing risks of unauthorized access over a long time, due to the token expiration policy.
- Supporting single sign-on while avoiding submission of the username and password strings in each request, which reduces risks of capturing the authentication information by intruders.

At the same time, this approach does not completely address security challenges of a distributed geo-processing grid-based system, due to the following reasons:

- Token passing is not a standard mechanism of web service security.
- Each service must be able to accept and process the security token coming as a parameter of the service request, which adds significant design overhead for the service developer.
- The numeric token does not contain any user credentials, and thus the user must be authenticated by the central server. This makes the security mechanism overly centralized and poorly scalable, and could become a bottleneck with the system growth.

2. A proposed solution based on GSI authentication

One of the main features of the GEON project mentioned above is integration of distributed geospatial information services, which, generally speaking, may follow different security policies and implement different security mechanisms. With ESRI's ArcWeb services being one of the web service providers accessible from GEON, we explored interoperatibility between ArcWeb's token based security system and GEON GSI based authentication system. The authentication scheme implemented in GEON, allows users to request a secure certificate from GEON Certificate Authority, which checks on an extensive set of user credentials before issuing the certificate. (Figure 2).

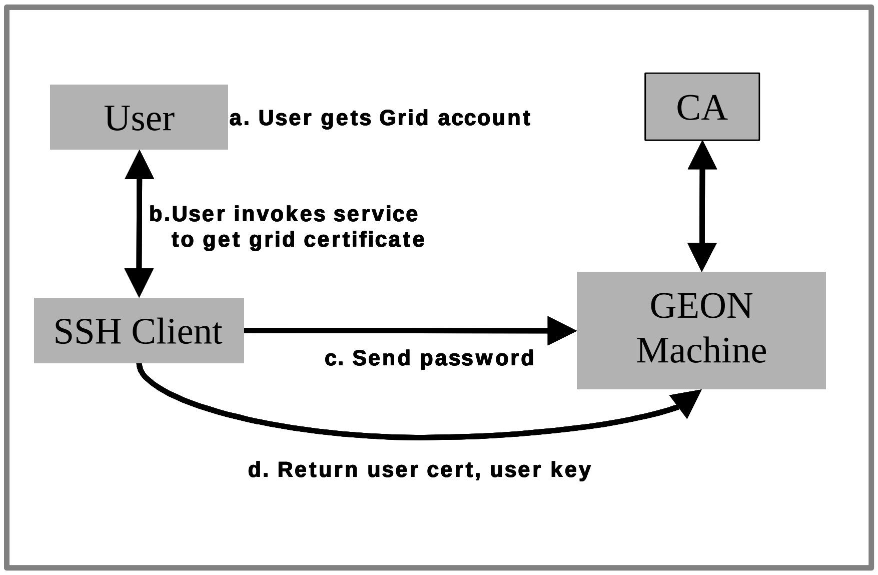
User
a. User gets Grid account
CA
b.User invokes service to get grid certificate
SSH Client
c. Send password
GEON Machine
d. Return user cert, user key

Figure 2: Obtaining the GSI accounts

Once the user has been approved by the Certificate Authority, UNIX accounts are created for this user/group of users on the computer running the CA engine. The user then needs to log in to the system to obtain its key pair (public and private key). This step is required only once. This public key, private key along with the user assigned pass phrase is used to generate a proxy certificate, which acts as a ticket to access the service. This certificate contains well defined user credential and uniquely identifies the user/group of users. This proxy certificate is also known as myproxy certificate among the community of GLOBUS Toolkit [7] users. To generate a myproxy certificate the user must invoke a service running on the myproxy server, which, in turn, considers the key pair and pass phrase provided by the user, as well as a grid map file that defines the list of users authorized by the CA. This mechanism is shown in Figure 3.

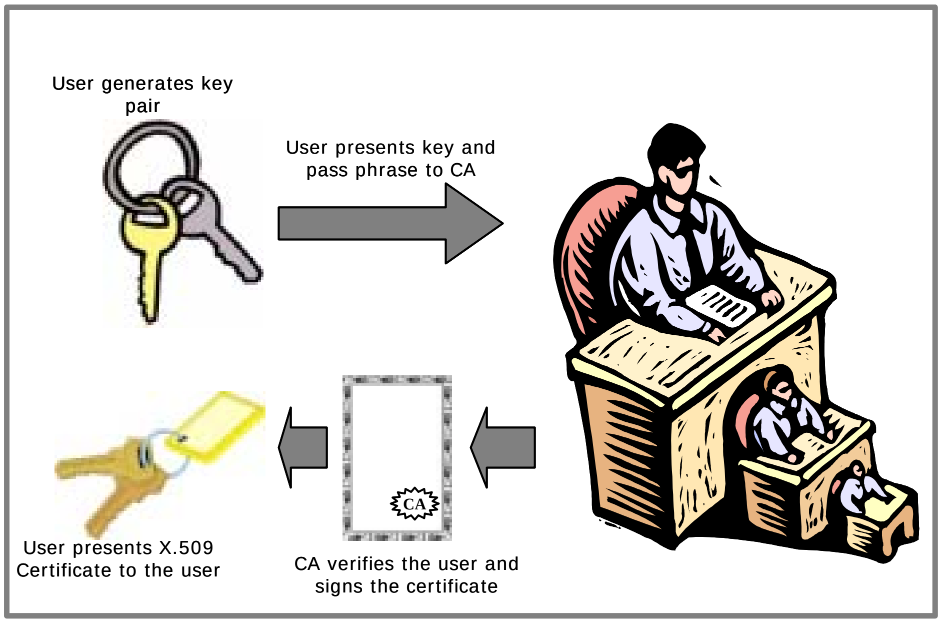
User generates key pair
User presents key and pass phrase to CA
CA
User presents X.509 Certificate to the user
CA verifies the user and signs the certificate

Figure 3: Obtaining X.509 Certificate

The steps involved in the creation of a proxy certificate are defined in figure 4.

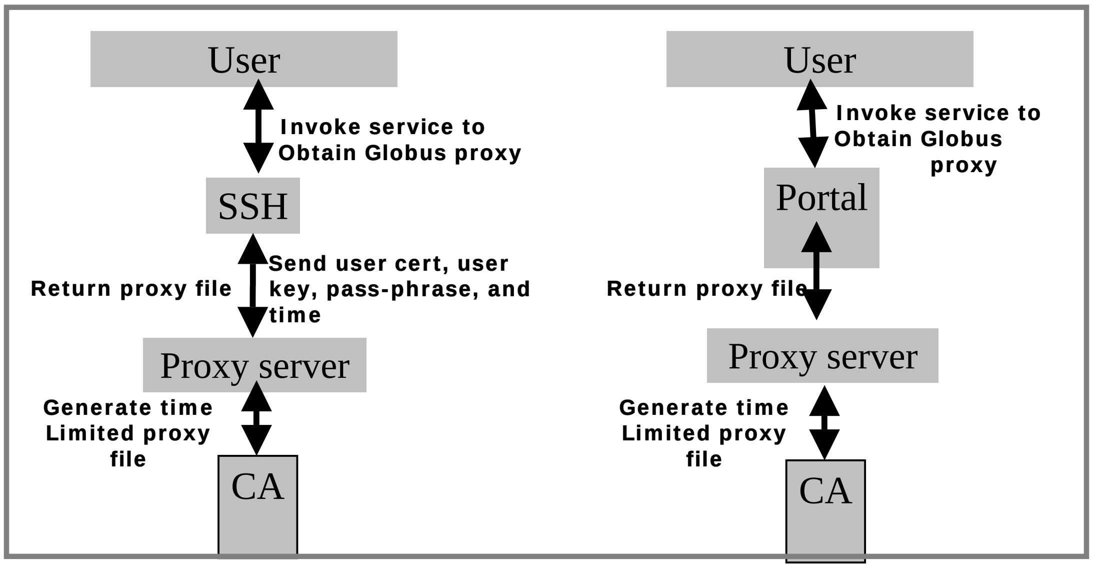
User
Invoke service to Obtain Globus proxy
SSH
Return proxy file
Send user cert, user key, pass-phrase, and time
Proxy server
Generate time Limited proxy file
CA
User
Invoke service to Obtain Globus proxy
Portal
Return proxy file
Proxy server
Generate time Limited proxy file
CA

Figure 4: Steps for creating myproxy certificate

In the first scenario, the user logs in to the myproxy server using a secure terminal session and executing command line scripts to obtain the certificate. The second scenario, designed for non-expert users, expects the user to log in to the GEON web portal and invoke a certificate generation portlet. In this case, the portal server takes the responsibility of running scripts for generating proxy certificates on behalf of users, and generates the output as a hyperlink or an email message sent. The security information defined in the proxy certificate is submitted as a part of the SOAP header, while the SOAP body carries the web service request itself. To access and extract the proxy certificate, we use the COG toolkit, which is a module in the Globus Toolkit. The extracted certificate string is then assigned to the SOAP header as a property using Java API for XML-based remote procedure calls (JAX-RPC). The Soap envelope appears as follows:

```
<SOAP-ENV:Envelope>
  <SOAP-ENV:Header>
       <mr:ContextInfo>
                Security certificate string
        </mr:ContextInfo>
     </SOAP-ENV:Header>
  <SOAP-ENV:Body>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

At the server receiving the request, the header must be decoded and the certificate must be extracted and validated. We extract the security information from the SOAP header using the JAX-RPC API, and decode it into a proxy certificate string. The certificate is then loaded into java objects defined in COG API for user validation. On successful validation access to the requested service is granted.
The GSI authentication described above is implemented in a wrapper over the existing ESRI authentication mechanism. Figure 5 illustrates how the myproxy certificate is mapped to the username and password in standard ArcWeb authentication. This wrapper module called *account mapper* is used to map accounts between GEON certificate users and ArcWeb users.

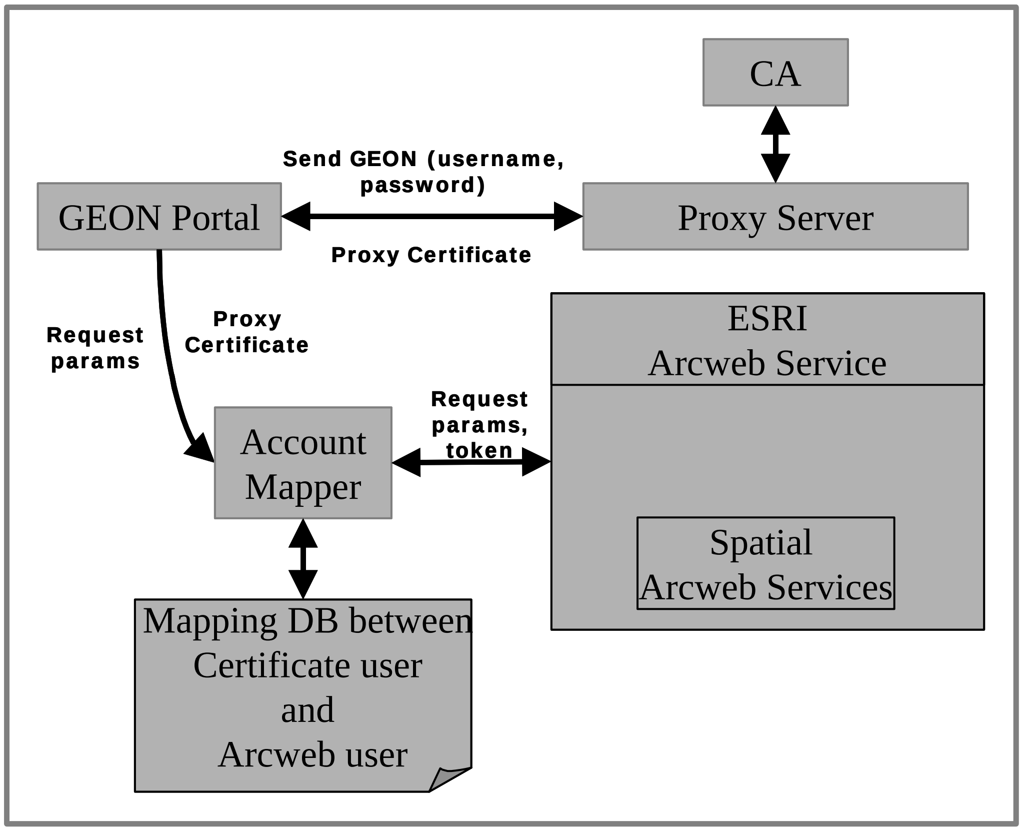

Figure 5. GSI Authentication for ArcWeb Services

Once a myproxy certificate user is successfully mapped to an ArcWeb user, an ArcWeb service is invoked. In the first step of this process, the Authorization ArcWeb service is invoked, which generates a numeric token based on the supplied username and password.. If the first step is successful, the actual requested ArcWeb service is invoked. The result of the service then propagates back to the user through the account mapper service. In the following section, we outline the advantages of the layered security mechanism (i.e. both GSI over ESRI token-based) for invoking ArcWeb services.

3. Decoupling authentication and web service development.

Decoupling of GSI authentication code from the rest of web service implementation makes it easier for web service developer who no longer has to be concerned about authentication issues and can concentrate on the functionality of the service. In our scheme (Figure 6) we maintain two generic components, namely the Authenticator Service running at the server and the Authentication Client running at the invoker's machine. At the service provider end, we configure the server in such a way that it invokes the authenticator before invoking the actual service. The authenticator looks up the defined security policies and user roles, and makes an authentication decision. All users roles, and access policies for each role, could be defined either in a database or in a flat file. A universal authenticator client at the requesting side then invokes the service on behalf of the actual client, adding the security certificate to the SOAP header of the requestor in the process. Relying on these standard authentication components, the programmer implementing a web service doesn't have to add any additional authentication code to the service.

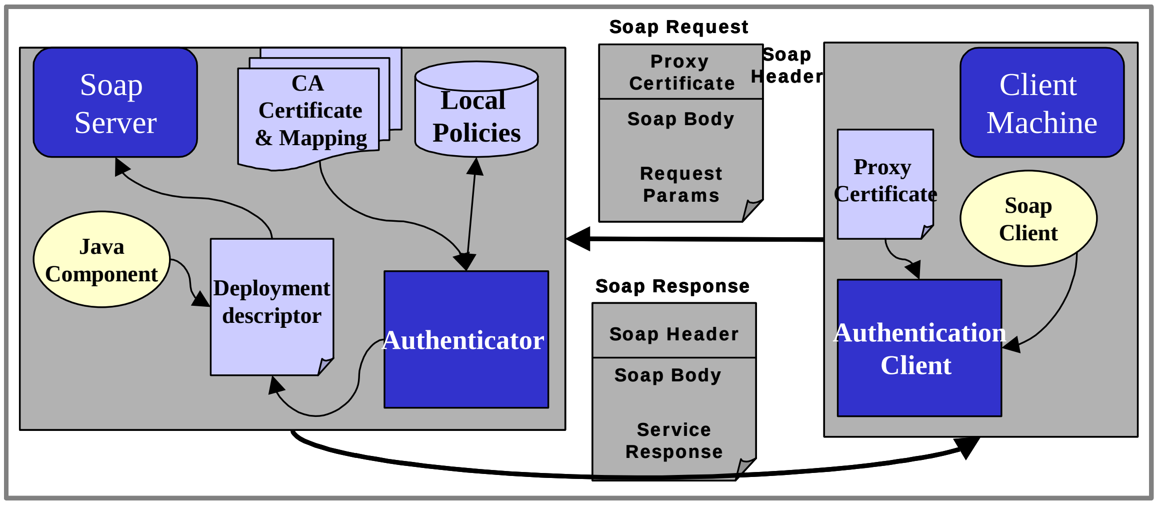

Figure 6. Decoupled Security Authenticator

Conclusion

GSI authentication complements the existing authentication mechanisms implemented in Arcweb services. Our approach provides for logical separation of the Meta information, like security and routing, from the actual request for invoking a spatial information service. Since the certificate passed between systems carries user credentials rather than a numeric token to identify specific user and its organization, the communication is more secure than standard token passing. With the proxy certificate acting like a token, we maintain a single sign-on capability while reducing the risks of password information being stolen. Having a predefined expiration policy for issued certificates makes the system less vulnerable to unauthorized uses. Since GSI works on mutual authentication, we get the benefits of distributed security across different organizations avoiding the potential bottleneck of a centralized security server.

Acknowledgement

"This material is based upon work supported by the National Science Foundation under Grant No. 225673" (note: 225673 is the grant number for GEON).

Thanks to Amit Behere (abehere@sdsc.edu) for his contribution during the implementation of the GSI authenticator.

References

[1] ArcWeb Services Overview
http://www.esri.com/software/arcwebservices/about/overview.html
[2] J2ME security, now and in the future
http://www.informit.com/articles/article.asp?p=30029&redir=1
[3] Securing J2ME wireless
http://www-106.ibm.com/developerworks/java/library/wi-secj2me.html
[4] X.509 Certificates and Certificate Revocation Lists (CRLs)
http://java.sun.com/j2se/1.3/docs/guide/security/cert3.html
[5] GSI Authentication
http://www-unix.globus.org/security/overview.html
[6] Using Java technology with GSI

http://www-106.ibm.com/developerworks/library/gr-ggsi/
[7] Globus Toolkit (GT)
http://www-unix.globus.org/toolkit/

Author

Ashraf Memon
Data and Knowledge Systems
San Diego Supercomputer Center
9500 Gilman Drive, MC-0505, La Jolla, CA-92093
amemon@sdsc.edu

Chaitan Baru
Data and Knowledge Systems
San Diego Supercomputer Center
9500 Gilman Drive, MC-0505, La Jolla, CA-92093
baru@sdsc.edu

Ilya Zaslavsky
Data and Knowledge Systems
San Diego Supercomputer Center
9500 Gilman Drive, MC-0505, La Jolla, CA-92093
zaslavsk@sdsc.edu

Steve Mock
Grids and Clusters
San Diego Supercomputer Center
9500 Gilman Drive, MC-0505, La Jolla, CA-92093
mock@sdsc.edu