



**University of Missouri-
Columbia**

Great Plains Network: Integrating Shibboleth, Grid and Bioinformatics

NMI-EDIT Case Study Series

In response to calls from the higher-education community, the NMI-EDIT Consortium has developed a series of Identity Management Case Studies to explore the planning and implementation of this critical infrastructure at higher-education institutions around the country.

In the spring of 2004, NMI-EDIT released the Extending the Reach Call for Proposal with the overall vision of exploring possible models for middleware support and informing the NMI-EDIT outreach and development efforts through collaboration with a wider, more diverse group of institutions. The work outlined in this case study was supported in part by the NMI-EDIT Extending the Reach Program.

This NMI-EDIT Case Study Series is sponsored by the National Science Foundation Middleware Initiative-Enterprise and Desktop Integration Technologies (NMI-EDIT) Consortium of EDUCAUSE, Internet2, and SURA. Additional support was provided by the National Science Foundation Cooperative Agreement NSF 02-028, ANI-0123937. Thanks are extended to authors Gordon K. Springer, Shafquat Bhuiyan, and Arturo Guillen.

Copyright © 2006 by The Curators of the University of Missouri. The Curators of the University of Missouri permits use of the content for noncommercial purposes with attribution. All rights reserved.



Executive Summary

This project is a supplement to the Great Plains Network Extending the Reach (ETR) project. It involves the integration of multiple middleware technologies to facilitate collaborative research and the sharing of resources in a multi-institutional, regional consortium.

The goal of this project is to deploy an application in the biological sciences to support researchers associated with the GPN/ETR project. It includes inter-institutional authentication via [Shibboleth](#)®, fine-grained authorization to access specific resources via entitlements, and provides access to three bioinformatics tools to utilize some commonly used genomic software, including a genomic database search capability. The system also employs grid computing technologies to utilize different resources available to the project.

The process of creating and deploying the application has been documented here to provide others with an example of application development as well as insights useful in deploying such applications in other

environments. In addition, the GPN/ETR group now offers a set of bioinformatics tools that can be shared among the GPN/ETR membership.

The section below lists some of the National Science Foundation Middleware Initiative-Enterprise and Desktop Integration Technologies (NMI-EDIT) components used in the described implementation. These components can serve as a base from which to grow and as a model for authentication and authorization utilizing readily accessible middleware technologies.

The results of this effort includes an operating test bed that is accessible by the GPN participants to gain secure access to shared resources among multiple institutions that can support collaborative bioinformatics and life science research in the region.

For more information on the methodology and implementation details described in this document, contact Gordon K. Springer at springer@missouri.edu.

NMI-EDIT Components Highlighted in this Case Study

eduPerson Directory Schema

<http://www.educause.edu/eduperson>

eduPerson contains identity-related attributes for higher-education institutions to deploy to foster inter-institutional collaborations.

Shibboleth System

<http://shibboleth.internet2.edu>

The Shibboleth System supports inter-institutional sharing of web resources that are subject to access controls.



Great Plains Network: Integrating Shibboleth, Grid and Bioinformatics

The Great Plains Network¹ (GPN) is a regional consortium of public universities in the states of Arkansas, Kansas, Missouri, Nebraska, North Dakota, Oklahoma, and South Dakota. The GPN Consortium has undertaken the development of a region-wide collaboration environment, utilizing standards-based middleware tools, to facilitate research and collaborative efforts throughout the region.

During March and April 2005, as part of the Great Plains Network Extending the Reach (GPN/ETR) project², a supplemental proposal was submitted to the National Science Foundation Middleware Initiative-Enterprise and Desktop Integration Technologies (NMI-EDIT) Consortium³ to fund a student to aid in the development of a prototype application that could help the GPN Consortium realize its goals of developing a region-wide collaboration and grid computing environment among its participating members. NMI-EDIT approved the supplemental request for support in May 2005. This report, in conjunction with the GPN/ETR Case Study report⁴, documents the prototype application implementation

process and provides a methodology and details for implementing similar applications elsewhere.

It is assumed throughout this document that the reader has read the GPN/ETR Case Study report³. In addition, it is assumed that the reader is familiar with and preferably has gone through the steps to implement a working system based on the NMI-EDIT Shibboleth® software⁵.

Project Description

Initially, the GPN/ETR project was to investigate and develop a deployment plan to utilize Shibboleth for inter-institutional authentication among member institutions in the GPN region. After participation by several project members at several Shibboleth InstallFests hosted by NMI-EDIT, Internet2 and the GPN itself, the project team was able to move beyond the investigation phase into the implementation phase. Due to the quick success in deploying Shibboleth at several institutions, the GPN/ETR group discussions turned to the development of applications that could operate in the group's multi-institutional environment and allow resource sharing among group members.

⁵ The Shibboleth Software is located at <http://shibboleth.internet2.edu>

¹ See www.greatplains.net

² See archie.csce.uark.edu/gpn

³ See <http://www.nmi-edit.org/index.cfm>

⁴ See the GPN/ETR Case Study report at: <http://archie.csce.uark.edu/gpn/publications/GPNCaseStudy.pdf>



One of the application areas of interest to the GPN/ETR participants was that of sharing resources to carry on research collaborations in the biological and life sciences. The primary goals of the prototype application project were the development of a collaboration environment and the investigation of the issues associated with deploying this type of resource across the regional GPN partner institutions. While most of the resources are presently housed at the University of Missouri, this is mostly a matter of convenience. The current implementation utilizes several separate computing systems and includes the means for controlling access to these resources via authorization entitlements. These entitlements are assigned to users through the Identity Management system and are verified whenever a user needs access to a particular resource.

Project Goals

The goal of this project is to deploy an application in the biological sciences to support researchers associated with the GPN/ETR project. It includes inter-institutional authentication via [Shibboleth](#), fine-grained authorization to access specific resources via entitlements, and provides access to three bioinformatics tools to perform some commonly used genomic tasks, including a genomic database search capability. The system also employs grid computing technologies to utilize different resources available to the project.

The process of creating and deploying the application has been documented here to provide others with an example of application development as well as insights useful in deploying such applications in other environments. In addition, the GPN/ETR group now offers a set of bioinformatics tools that can be shared among the GPN/ETR membership.

Methodology

[Shibboleth](#) software has been deployed to restrict access to all of the GPN/ETR project resources to authenticated users and access is further limited by project-defined entitlements to specific resources. Researchers with the proper authentication and authorization entitlements can access their data via a web interface that utilizes parallel and distributed computing techniques behind the scenes to perform computationally intensive tasks. Using a distributed grid across resources in the GPN, it is possible to harness the project's computing power in a controlled, authenticated and authorized way.

Starting with a base [Shibboleth](#) installation, the process used to incorporate the bioinformatics tools was to get the base tool software installed, create a Web-based interface to execute the tool from the Web, and then add the required authorization code to the Web interface to insure the proper authorization steps were followed to gain access to the respective tools.

A synopsis of what is required to authenticate and authorize access using



Shibboleth and entitlements are listed in Appendix A. A flow chart of the authentication and authorization processing is provided in Appendix B. By focusing primarily on the last two boxes in the flowchart (“User has authorization attributes to access the resource?”, “If yes, user is granted access.”), this case study addresses how to take an application and incorporate it into a Shibbolized environment.

One of the initial GPN/ETR target applications was deployed at the University of Missouri-Columbia (MU). The deployment plan included utilizing an existing, Web-based application that supports several genomics projects and has a locally developed authentication mechanism that restricts access to the genomic web site to authenticated users. For the GPN/ETR project, this existing application’s locally developed authentication front-end was removed and was replaced with the **Shibboleth** authentication process, thus **Shibboleth** would now be used to access the target application.

However, replacing the initial authentication step for accessing the genomic application did not replace the security mechanisms built throughout the application itself, including file upload and download capability in both private and group repositories, the ability to change a user’s individual password and the ability to restrict access to research data. To integrate with these mechanisms, a special login from the main web page was created

and is displayed after a **Shibboleth** authenticated user logs into the target (Service Provider) site. This login performs an internal login to the original application using the **Shibboleth** user’s credentials obtained from the Identity Provider. This modification preserved the original application’s security mechanisms and services without requiring any changes to the application (except for replacing the initial authentication process).

Once logged into the Service Provider (target) system, the user has all links and services provided by the Service Provider potentially available to them. However, this application was being deployed in a multi-institutional collaborative environment and since the intention was not to grant anyone who can authenticate with their home institution’s Identity Management system access to all of the services available from the Service Provider, a finer-grained authorization for determining a user’s authorization to access specific resources was needed. After much discussion, the Entitlement attribute in the NMI-EDIT **eduPerson**⁶ data structure was chosen to provide the necessary authorization granularity the project’s prototype genomics application would require for successful deployment across the GPN. The Entitlement attribute contains the list of entitlements (authorization values) each user has assigned to them when they authenticate through their home institution

⁶ The eduPerson Schema is located at <http://www.educause.edu/eduperson>



(see Entitlements section below). These entitlement values are validated by the local application to determine if an authenticated user has the necessary authorizations to request access to the different services offered at the target web site.

Using [Shibboleth](#) authentication and the user's defined [eduPerson](#) entitlements (obtained from the authentication), the target server and the target application can determine if the requesting user is authorized to make the request for service (e.g. upload a file, execute a grid application).

Deliverables

The deliverables from this genomic application deployment project are the provisioning of a working application in bioinformatics that can be shared by researchers across the GPN region. In addition, documentation is provided that details the application implementation process and the use of entitlements to provide fine-grained authorization control over application access. In particular, the deliverables include:

- A working version of an integrated bioinformatics application that utilizes [Shibboleth](#) for authentication, [eduPerson](#) entitlements for authorization and grid computing resources for computational results.

- Provide a case study on creating, deploying and managing the [eduPerson](#) Entitlement attribute for authorizing access to computing resources in a distributed, virtual organization environment.
- Documentation for users and developers.

Entitlements

Entitlements are the values of the Entitlement attribute in the [eduPerson](#) object class definition. In the GPN/ETR project, the namespace *mace:urn:greatplains.net* was registered with the Internet2 Middleware Architecture Committee for Education (MACE). The Universal Resource Name (URN) registry⁷ is used to register persistent names to resources. The GPN/ETR project uses these names not only to identify resource names, but also as authorization values that, if a requesting user has the appropriate resource tied to their identity information as an entitlement, are then used to access these resources.

A detailed description of the entitlements used by the GPN/ETR group is contained in the GPN/ETR Case Study report⁸. These details are not repeated here. The entitlement descriptions for the *greatplains.net* delegated namespace can be obtained from the registered namespace web site⁹.

⁷ See middleware.internet2.edu/urn-mace

⁸ See the GPN/ETR Case Study report at: <http://archie.csce.uark.edu/gpn/publications/GPNCasestudy.pdf>

⁹ See <http://www.greatplains.net/mace-gpn>



For the GPN/ETR project, the **eduPerson** Entitlement attribute must be set for each individual to be authorized in order for that individual to be able to use any of the GPN **Shibboleth** resources. Since the individuals are located in multiple institutions, the Identity Management System (IDMS) at each of the participating institutions must incorporate the GPN entitlements into their individual IdP systems. This is necessary since the entitlement information must be released with an individual's authentication information as part of the **Shibboleth** authentication process. The result of this process is that the Service Provider (SP or target) that initiates the request to authenticate is assured that the user has successfully authenticated and receives a set of entitlement values to be used to authorize access to all, some or none of the services located at the SP. In addition, the decision to grant or deny access is under control of the SP at the application level itself.

Since the GPN/ETR SP at MU is utilizing the **Shibboleth** authentication and the entitlement attributes in conjunction with another layer of security imposed by additional, locally constructed security mechanisms, it would be extremely difficult to provide a template for building applications under **Shibboleth** and entitlements for easy duplication elsewhere. Therefore, the GPN/ETR project team modified the authentication mechanisms such that the Biotools application requires only the **Shibboleth** authentication, the greatplains.net entitlements and the validation of the credentials in each of the developed

applications. Whereas the other applications in the MU SP still operate with the third level of MU locally-developed authorization. The Biotools application is self-contained to use only the **Shibboleth** authentication and the **eduPerson** entitlements. This fact demonstrates the flexibility of the prototype application deployment methodology to permit varying degrees of security and protection at the discretion of the SP and the applications themselves.

A key part of the modifications was to create a "session" database that keeps track of the users as they login to the SP and to keep track of the user's authorization attributes so they can be used by the SP application servers whenever needed. In addition, this session database prevents anyone from trying to "replay" a session without successfully logging into the system through **Shibboleth**. That is, a user cannot keep track of a URL used during a session and simply specify the URL in their web browser at a later time and expect it to work. User credentials are purged from the session database when they logout of the SP or are idle for a period of time by not moving between application web pages. In either of these cases, the user will have to re-authenticate with **Shibboleth** before being granted access to the SP services again.

Shibboleth Entitlement Management and Session Tracking

Shibboleth focuses on the authentication of users trying to access the services at a target web site (SP). The authentication process



redirects the incoming request to an appropriate Identity Provider to allow the user to authenticate with the IdP that can verify the user's authentication information. If successful, the original request to access the SP is redirected back to the SP with [Shibboleth](#) generated credentials set to permit the access to the target site. However, [Shibboleth](#) itself does not provide support for web application developers to authorize users with fine-grained access control. Without authorization control, an authenticated user would have access to all services provided by the SP. Authorization control is needed when a SP offers multiple services that require different authorizations to access the different services. To create fine-grained authorization control, entitlements were used to discriminate among the services provided to individual users making requests. Also implemented was a simple, but strong session tracking mechanism that can take advantage of the [Shibboleth](#) authenticated environment and use the entitlement information for validating a user's authorization privileges to access specific services at the SP. This mechanism works even for non-[Shibboleth](#) aware web applications.

Once a user has authenticated using [Shibboleth](#), a number of [Shibboleth](#) generated web server environmental variables are set. These are a user's [Shibboleth](#) credentials that can be used by the target web server and application programs as an authorization mechanism as needed. In particular, the HTTP_SHIB_EP_ENTITLEMENT variable is

used throughout a session to validate a user's privileges to access different services offered at the SP. The [Shibboleth](#) generated variables provide critical information about the user, which is used for authorization purposes. In order to effectively track a user, all the necessary [Shibboleth](#) generated information is available in an easily accessible, but secure place. A database table was set up in the local Oracle database to store this information. The schema for the table is as follows:

```
SQL> desc shib_entitles;
```

Name	Type
NAME	VARCHAR2(128)
ENTITLEMENT	VARCHAR2(1024)
SID	VARCHAR2(512)
TIME_STAMP	VARCHAR2(128)
LAST_ACCESS_TIME	VARCHAR2(128)
IP_ADDR	VARCHAR2(64)

For this project, this set of data provides the needed information. The [Shibboleth](#) generated session identification number is used as the primary key in the database. TIME_STAMP stores the time when the session was created. LAST_ACCESS_TIME keeps the time when the user last made some request (event). This facilitates the prevention of a replay attack and also allows us to time out a user after a predetermined period of time instead of waiting for the user to close his browser. In this case the time was set out to a period of twenty minutes.

After the initial [Shibboleth](#) authentication and the user accesses the index page, the user's [Shibboleth](#) credentials are recorded in the Oracle database. The [Shibboleth](#) session id is then passed to any subsequent page. With



just the session id, the code that is called to dynamically generate a web page can choose to use any of the [Shibboleth](#) credentials for authorization purposes. To do this, a set of functions was developed that efficiently checks a user's credentials and entitlements. All of these functions are located in the `checkEntitle.php` page. In addition, a standard block of code was developed that can get the session id passed to a page and do the authentication and authorization with minimal modification. This enables the web application developer to make his pages [Shibboleth](#) aware with very little effort. A sample of the block of code is listed in Appendix D.

The only customization a developer needs to do is provide the required entitlement string a user needs to have to access this page. For even more simplicity and scalability he can modify this block once and put it in a separate file and include it in all of his pages, if all these pages require the same entitlement. The `checkEntitle` function checks whether the session id has expired or not and if not whether the user has the required entitlement. The `updateTS` function updates the `LAST_ACCESS_TIME` with the current time as the users last active time. The `getEntitle` function returns the user information without checking the time out. This simple mechanism allows the tracking of a user during their [Shibboleth](#) session. This tracking can also be done remotely as long as a remote web application has access to the session database. Three web applications were customized (two of them are in the same

system as the database and one is hosted in a different system) using this tracking mechanism to provide fine-grained access control based on the user's entitlements. Thus, these applications have been made [Shibboleth](#) aware with little modification.

A session management page has also been developed for administrative purposes. It shows the currently active or recently expired [Shibboleth](#) sessions recorded in the Oracle database. This management page can also purge old sessions that have expired. Note that an expired session entry in the database cannot be used to gain access to the SP services. A user with an expired session entry must re-authenticate with [Shibboleth](#) before services will be provided again. For demonstration purposes, a sample management page has been made available¹⁰.

Application Development

The application development process is documented for the respective bioinformatics tools that are included in this project. The tools are: Clustal W (a global multiple sequence alignment program), Blast (a genomic database search package), and mpiBlast (a parallel version of Blast designed to run on grid computing clusters). These tools were chosen since the source code for them is readily available for download from the Web. Each of the programs can require substantial computing resources (both storage and computation) and are frequently used by

¹⁰ See genome.rnet.missouri.edu/Biotools/php-scripts/shib_sessionDB.php



researchers as part of their laboratory and research activities.

A set of step-by-step installation procedures is provided for each package and is available on the Web from the GPN/ETR Service Provider located at the University of Missouri-Columbia¹¹. The installation steps include installing the native application from the download and the needed steps to incorporate the application into the Shibboleth environment.

Bioinformatics Components

Clustal W

Overview of the Software

Clustal W is a general purpose multiple sequence alignment program for DNA or proteins. The Clustal W source code can be obtained from the European Bioinformatics Institute¹² using a Web browser to download the source for different computing platforms. The Clustal W 1.83 version of the code was used for this project.

Clustal W is used to produce biologically meaningful global multiple sequence alignments of divergent sequences. It calculates the best match for the selected sequences, and lines them up so that the identities, similarities and differences can be seen.

Installation and Training Issues

Since Clustal W is designed to be an interactive program, it was necessary to build a Web interface that could execute the program in a non-interactive manner and to convert the resulting output to a standard HTML output format in this application. In addition, Clustal W produces auxiliary output files during the analysis phase; one file being a phylogenetic tree (showing the “distances” one sequence has from the other sequences being analyzed at the same time). This phylogenetic tree is incorporated into the analysis output by using a locally modified version of the NJPLOT program obtained from the Université Claude-Bernard Lyon¹³

WWWBlast

Overview of the Software

One of the most widely used programs in bioinformatics is a program named BLAST. The Basic Local Alignment Search Tool (BLAST) finds regions of local similarity between sequences. The program compares nucleotide or protein sequences to sequence databases and calculates the statistical significance of matches. BLAST can be used to infer functional and evolutionary relationships between sequences as well as help identify members of gene families. WWWBlast is a standalone program that is part of the National Center for Biotechnology Information (NCBI) software toolkit. This program performs requested BLAST searches against a variety of public and private databases formatted for high-speed searches

¹¹ See crick.rnet.missouri.edu/Biotools/Doc

¹² See ftp.ebi.ac.uk/pub/software/unix/clustalw

¹³ See pbil.univ-lyon1.fr/software/njplot.html



of genomic data. BLAST and its ancillary programs can be obtained from the NCBI FTP site.¹⁴

After downloading, compiling and configuring the software for local, standalone usage, both the code and the generated HTML output were customized to operate in the environment being developed for the GPN.

Installation and Training Issues

After compiling the NCBI Toolkit code, a Web interface was built to validate the user's authorization to access the program, collect optional runtime parameters governing the database(s) to be searched, the sequence(s) to be analyzed and search-time options required to run the program. After the submission of the request, the DB search is performed by BLAST and the resulting HTML output is post-processed to incorporate local customizations. Among these customizations is the inclusion of HTML links to other data associated with the results such as go to the NCBI Web site to view a particular sequence found during the DB search.

WWWBlast is a standalone program that uses a single multiprocessor, high-performance computing (HPC) system to perform the database search. In our case this is an HP Alphaserver SC cluster system located at the University of Missouri. This system supports both high-volume genomic projects at MU as well as researchers performing individual analysis tasks. And, the jobs submitted through the GPN/ETR **Shibboleth** interface are

¹⁴ See ftp://ftp.ncbi.nlm.nih.gov/toolbox/ncbi_tools

accomplished just as if the requesting user were a local MU user with an authorized userid obtained from MU. However, the authorization step occurs through the **Shibboleth** environment by using a locally written "shiblogin" program that verifies the user's **Shibboleth** credentials and entitlement authorizations before granting access to the BLAST program.

mpiBlast

Overview of the Software

mpiBLAST is a freely available¹⁵, open-source, parallelization of NCBI BLAST. mpiBLAST segments the BLAST database and distributes it across cluster nodes, permitting BLAST queries to be processed on many nodes simultaneously. mpiBLAST is based on MPI and runs under Linux, Windows, and various varieties of Unix. mpiBlast originated at the Los Alamos National Laboratory (LANL).

Installation and Training Issues

In order to install mpiBlast, both the NCBI Toolkit source and the modifications to the NCBI code provided by LANL are required. In addition, if not already installed, an appropriate version of either the MPICH¹⁶ or LAM/MPI¹⁷ message passing interface software must be installed on the target cluster system. In this case, the LAM/MPI code was installed for use by the mpiBlast program.

The project team utilized a small Dell Xeon PowerEdge 1850 cluster at MU to validate the

¹⁵ See mpiblast.lanl.gov and <http://www-unix.mcs.anl.gov/mpi>

¹⁶ See <http://www-unix.mcs.anl.gov/mpi/>

¹⁷ See <http://www.lam-mpi.org/>



operation of the installed code before transferring it to a large cluster for more productive use.

Installation Procedures

In order to create the desired set of Web pages to support the GPN/ETR project, each of the software packages were obtained from the respective distribution sites. These packages were then installed on the systems where the programs need to reside to provide the respective services. Once each of the packages was installed, they were tested in their native implementation to be certain of their proper operation.

Once the installations were verified, a Web interface was developed for access to each of the programs from the Web. As part of the development, Common Gateway Interface (CGI) programs were created that could accept input from users via the Web, form an appropriate request to execute the respective programs on the respective systems where the programs reside, execute the request and retrieve the output generated by the request and create or modify the results to produce an HTML formatted Web page containing the results. The resulting HTML output is then returned to the user at the conclusion of the CGI request. The developed programs, HTML pages and CGI programs to perform the respective analyses are summarized in

Appendix D. A downloadable Tar file with this code is also available¹⁸

Discussion

This document and the GPN/ETR Case Study report¹⁹, along with the cited URLs, describe the process of developing applications that can be deployed at a [Shibboleth](#) Service Provider. The process is quite straightforward once the basic components of [Shibboleth](#) are in place. If fine-grained authorization processing is not required, then application deployment can be as simple as defining web pages at the Service Provider (SP) web site. However, in a multi-institutional environment or virtual organization (VO) environment, finer-grained access control is necessary. In particular, in environments such as the GPN's where large-scale resources are not to be made available to the community at large, the use of entitlements is required in order to provide a more discriminating access control mechanism.

This prototype application deployment project has shown that the process of building such fine-grained discrimination into an application environment need not be an overwhelming task. With a planned set of entitlement values and the use of available middleware software, a standards-based, secure environment can be created while leaving access control in the hands of the SP, even when multiple

¹⁸ See Biotools Source Files at <http://genome.rnet.missouri.edu/Biotools/Doc/Biotools.tar.gz>

¹⁹ See the GPN/ETR Case Study report at: <http://archie.csce.uark.edu/gpn/publications/GPNCasestudy.pdf>



institutions are collaborating with a shared set of resources.

Time did not permit the GPN/ETR project team to address a major, unsolved problem - creating and administering the entitlements across multiple Identity Providers (IdP). At the outset of this project, the team hoped to work with the Internet2 Signet group to investigate the use of their administration tools in the GPN's multi-institutional environment. Unfortunately this did not happen, but the GPN/ETR team plans to continue working with the Signet group on this problem in the future.

The primary hurdle in creating and administering the entitlements across multiple IdPs involves the ability to "push" entitlements into a foreign IdP's Identity Management (IDM) system. Deploying this "push" requires the resolution of numerous issues related to who is authorized to push changes into what IDM, what agreements and trust relationships are needed before any action can occur, and how are such changes performed to update a wide variety of different IDM system implementations. Since institutional policy typically prevails for making such decisions, collaborations will be impeded unless appropriate agreements between and among institutions can occur.

The methodology at present is to side-step this issue by creating a collaboration environment as an autonomous entity. The main drawback of this methodology is easily demonstrated by asking the question: "How many separate user

ids and passwords should each individual have to keep track of on a daily basis?" While, for a variety of security reasons, it is not desirable to have only one user id, it seems overwhelming to both users and system administrators for users to have to have hundreds of user ids - one for each separate task they need to do. Striking the correct balance in the number of ids while maintaining security is an incredibly complex issue, but one that must be resolved.

In Conclusion

The initial motivation for this project was to create a tool that could serve as a model to carry out large volume searches of genomic databases in a shared, grid computing environment. In order to do this, the project team needed to create a shared collaboration environment and build the means for accessing the environment's shared resources in a secure and controlled way. Along the way the team dealt with a large number of issues, each with their own complexities. The project has resulted in the beginnings of a tool set that includes two different means for running Blast searches and a third tool that performs multiple sequence analysis. These programs presented similar, but different, challenges in order to deploy them in the project's environment. Nevertheless, the GPN/ETR team has demonstrated the feasibility of not only building Shibboleth applications, but that of utilizing shared resources across a grid computing environment.



This document describes the methodology and steps required to implement the set of bioinformatics tools under a [Shibboleth](#) authenticated environment. In addition, it discusses the use of [eduPerson](#) entitlements to provide a way to support fine-grained authorization control over the specific services offered at a particular [Shibboleth](#) Service Provider site. The ability to provide access controls is critical in a shared, collaborative environment since different groups of collaborators may share the same SP resource, but not all users should necessarily be granted access to all of the resources. Without a means for discriminating means for determining user access, a great deal of additional, and redundant effort is necessary, since separate [Shibboleth](#) targets must be created for each separate collaboration group within the same virtual organization. Utilizing entitlements allows an organization to avoid having separate targets and lets authorized users access the resources they are entitled to access while preventing them from accessing resources they are not entitled to use.

Over the life of the GPN/ETR project, the project team met regularly to discuss common [Shibboleth](#) deployment issues and held seminars to educate each other in the technologies being utilized in the project. Some of these presentations, as well as those made at the GPN Annual Meeting, the Internet2, EDUCAUSE, and NMI-EDIT national meetings provide materials about the [Shibboleth](#) authentication, authorization and grid computing discussed in this Case Study, have

been made available²⁰. The reader is encouraged to look at these presentations for further clarification of the material presented here. These presentations include some actual screen shots of the process that takes place to authenticate and authorize a user to access various resources.

Since the code that has been provided can consume significant computing resources, it is necessary to restrict access to users who can authenticate into the project's [Shibboleth](#) environment. If just anyone was granted access to the site, not only would a door have been opened for abuse of resources, but also the purpose of creating a secured environment with appropriate authorizations would have been defeated as well. The GPN/ETR team, however, accepts requests from anyone wanting to gain access to their [Shibboleth](#) Biotools site²¹ at the following email address: springer@missouri.edu. By providing access to a short-term userid and password that can be used to authenticate with the [Shibboleth](#) authentication process, the GPN/ETR generously enables others to see the system in action.

The project team set out to build a bioinformatics application that combined the mechanisms of [Shibboleth](#) authentication with, entitlements for authorization and would utilize grid computing resources without significant administrative overhead. This has been accomplished and the code used to create the application has been documented and made

²⁰ See <http://genome.rnet.missouri.edu/GPN/Docs>

²¹ See <http://crick.rnet.missouri.edu/Biotools/>



available to others as a model. It is hoped that by doing this, others will be able to implement similar environments elsewhere and further their ability to collaborate and share resources in a straightforward manner. The prototype applications deployment provided the GPN/ETR team members involved with the project with a rich learning experience and will likely do the same for others that undertake a similar project in their own environment.

There are a number of similar kinds of biological applications that can employ the same techniques to distribute computations across a grid-enabled environment. Thus this project serves as a model others may build on in order to provide the ability to authenticate, authorize and distribute computations in an organized and secure manner. In this way, this prototype application deployment project extends the reach to other's research environments and offers a proof of concept for the methodology being developed.

More Information

For more information on the methodology and implementation details described in this document, contact Gordon K. Springer at springer@missouri.edu.

Acknowledgements

This work was supported, in part, by IATS/Research Support Computing at the University of Missouri-Columbia (iatservices.missouri.edu/research) and through an NMI-EDIT Extending the Reach grant (<http://www.nmi-edit.org>) to The Great Plains Network. The endless hours Shafquat Bhuiyan and Arturo Guillen put in to make the proposed design work were invaluable. There are also a number of people who contributed to this project at earlier stages including Larry Sanders, Denis Hancock, Mike Woodson, and Abdul Khambati. Thanks go to them all for their efforts that made this project both feasible and successful.



Appendix A: Synopsis of the Processing

What Do We Need To Do?

- ❖ Programs and data need to be shared across institutional boundaries.
- ❖ Authorized persons need appropriate access to the shared programs, data and resources.
- ❖ Unauthorized persons need to be excluded from access.
- ❖ We need to be able to trust other institutions to handle authentication and authorization properly.
- ❖ User privacy must be maintained consistent with applicable laws and individual user preferences.

How do we do it?

- ❖ Manage authentication and authorization in a consistent way between institutions.
- ❖ Agree upon a set of attributes that allow access decisions to be made.
- ❖ Employ existing open technologies (LDAP, [eduPerson](#) schema, [Shibboleth](#), Apache, Tomcat, SAML, etc.)
- ❖ Develop a trust relationship between all members of the Virtual Organization.
- ❖ Create a web-based data repository.

What is a Federation?

- ❖ A Federation is defined by agreed-upon policies.
- ❖ A federation builds a network of trusted organizations so an individual does not have to negotiate with all the other members for local user accounts.
- ❖ A single, trusted certificate authority eliminates the need to self-sign SSL certificates, thus achieving a higher level of security.
- ❖ A Virtual Organization can use the federation structure to handle the inter-institutional needs for trusted authentication.
- ❖ InQueue (inqueue.internet2.edu) is a test federation designed to assist in learning how federations work.
- ❖ InCommon (www.incommonfederation.org) is designed to be a production federation. Planning is underway to utilize InCommon for production use in the GPN/ETR project.

How this works

- ❖ The Identity Provider (IdP) authenticates the user and releases attributes to the Service Provider (SP).
- ❖ The SP authorizes the user, based on attributes released by the IdP.
- ❖ [Shibboleth](#) (shibboleth.internet2.edu), along with a federation, provides the framework in which full trust can be achieved among organizations.



Attributes

These are typically defined in an LDAP directory. The [eduPerson](#) schema (www.educause.edu/eduperson) is added to the standard LDAP schemas to provide additional attributes appropriate for educational institutions.

- ❖ The IdP determines its attribute release policy (ARP), taking into account federal and state law, institutional policies, owner preferences, etc.
- ❖ The SP determines its attribute acceptance policy (AAP), taking into account institutional/security needs.
- ❖ The eduPersonPrincipalName, eduPersonEntitlement, and eduPersonAffiliation are required for access to the Great Plains Network Data Repository.
- ❖ The eduPersonEntitlement consists of a URN or URL that asserts a privilege for a user. For example: urn:mace:greatplains.net:repository
- ❖ The urn:mace:greatplains.net namespace has been assigned to the Great Plains Network by the Middleware Architecture Committee for Education (middleware.internet2.edu/urn-mace), and the namespace below greatplains.net is documented at www.greatplains.net/mace-gpn
- ❖ The SP generally filters these based on various criteria, thus retaining control over which attributes are accepted, and from whom.

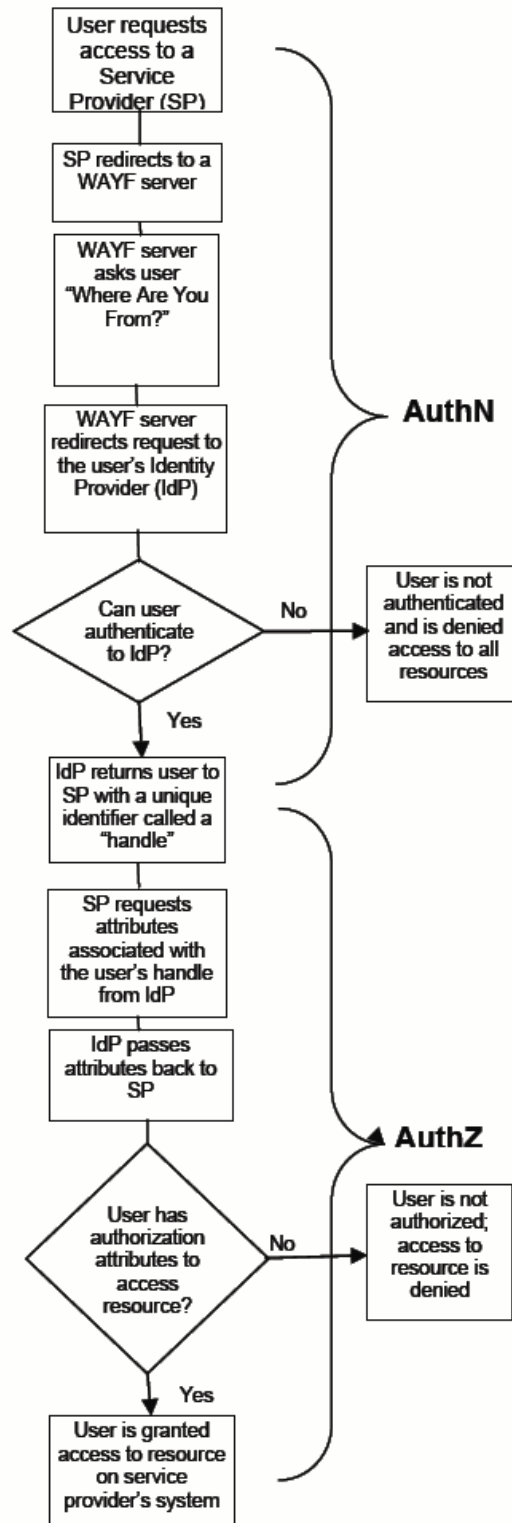
With proper planning, a hierarchy of privileges can be established, giving the SP fined-grained control over access to resources.

Summary

Using a set of agreed-upon [eduPerson](#) attributes, the Research Support Computing group at the University of Missouri -- Columbia has created a framework for a data repository and grid-based applications shared among members of the Great Plains Network (crick.rnet.missouri.edu/GPN).



Appendix B: AuthN/AuthZ Flow



Appendix C: Glossary

- ❖ **Attribute** – A piece of information about a user.
- ❖ **AuthN** – The authentication of a user.
- ❖ **AuthZ** – The authorization of a user.
- ❖ **Entitlement** – An attribute in the [eduPerson](#) Object Class whose value is used for fine-grained authorization by the GPN/ETR project.
- ❖ **GPN/ETR** – The Great Plains Network Extending the Reach project (funded by NMI-EDIT) that supported the project supplement described in this document.
- ❖ **IdP** – Identity Provider (usually the user’s home institution); also known as “Origin.”
- ❖ **LDAP** – Lightweight Directory Access Protocol.
- ❖ **SAML** – Security Assertion Markup Language.
- ❖ **SP** – Service Provider; also known as “Target.”
- ❖ **WAYF** – “Where are you from?” A server normally associated with a federation that accepts redirects from an SP and redirects to an IdP.
- ❖ **PHP** – General purpose scripting language that is especially suited for Web development.
- ❖ **Tarball** – An archive of files created with the Unix tar utility.



Appendix D: Description of Source Code

❖ The Source Code

A tarball file named `Biotools.tar.gz` that contains all the code to run the three Shibbolized applications: Clustal, MPIBlast and WWWBlast will be provided. You should place the `Biotools.tar.gz` somewhere under the `DOCUMENT_ROOT` of the web server where the applications are to be installed. Having done this, the code will be unpacked and the following directory tree structure will be accessible:

```
./Biotools/  
  /Clustal <- Clustal W code  
  /Doc <- Documentation (can be accessed via web)  
  /WWWBlast <- Web version of blast  
  /images <- images referenced in the scripts  
  /mpiblast <- MPIBlast distribution  
  /php-scripts <- "library" of PHP scripts (for authentication, connection to DBs, etc)
```

There are README files in several of these directories that should be read for details on each utility requirements, configuration steps and code description.

If the proper [Shibboleth](#) environment and all the requirements specified in the README files are fulfilled, then pointing a browser to the Biotools directory with the right entitlement by an authorized user will bring up a menu with options to run the applications.

❖ Authentication and Authorization Code

To facilitate authentication and authorization from any program we have developed a set of PHP functions that can be called from any script and verify user access. A brief description of these scripts can be found below. For more technical detail please look at the source code provided with this report. The code is fairly readable with useful comments.

➤ The `checkEntitle.php` script:

This script contains three major function definitions, `checkEntitle`, `getEntitle` and `updateTS`. Any script that wants to check a [Shibboleth](#) authenticated user's credentials can call these functions and verify. `checkEntitle` takes a [Shibboleth](#) session ID, the user's IP address and an entitlement to verify and returns true if the user has it or returns false otherwise. `getEntitle` takes a [Shibboleth](#) session ID as parameter and returns entitlements of the user associated with the session ID. `updateTS` updates the last activity time of a user. Any PHP script can include this file and take advantage of these functions. In fact we have used a standard block of code using these functions of all of our scripts. It can be found below.



```

<?php
/* Get the Shibboleth session ID */
if(!isset($_REQUEST["shibsid"])) {
    $sid = "";
} else
    $sid = $_REQUEST["shibsid"];

/* Check required entitlement to access this page */
require_once('checkEntitle.php');
if(checkEntitle($sid, $_SERVER['REMOTE_ADDR'],
    "*** REQUIRED ENTITLEMENT TO VIEW THIS PAGE ***")) {
    exit;
}

/* Update last access time */
updateTS($sid, $_SERVER['REMOTE_ADDR']);
/* Get user information from database */
$entitleInfo = getEntitle($sid, $_SERVER['REMOTE_ADDR']);
$entitleInfo = explode(" ", $entitleInfo);
$entitle = $entitleInfo[0];
$shibsid = $entitleInfo[1];
$ts = $entitleInfo[2];

?>

```

The only customization that needs to be done in this block is placing the required entitlement to access a certain page in checkEntitle function call.

➤ **The connection.php script:**

This file contains the Oracle database connection function. One willing to use this file needs to provide the user name, password and Oracle host information in the oci_connect function.

➤ **The enterShibEntitle.php script:**

This is the script that gets called every time a user authenticates through a [Shibboleth](#) environment. It takes all the user information and enters it to Oracle session database. If any previous record of the user logging in from the same IP address is found, it gets overwritten. Otherwise a new session is recorded in the database. This script is particularly helpful if your [Shibboleth](#) service provider machine has no local or remote access to Oracle database containing the session database.

❖ Clustal W Code

Once the clustalw and the njplot programs are installed on the target system (see [./Biotools/Clustal/README](#) file on the tarball for instructions on where to find and install these utilities) we need to create an interface to them. To do so we have written a series of PHP scripts to interact with the Clustal W application through the user's web browser. It is assumed that a PHP enabled web server is running on the target machine. Here is the list of the scripts (under [./Biotools/Clustal](#) directory):



➤ **The index.php script:**

This is the first script and is the GUI of the program. These scripts interact with the user to get the user's input through the following available options:

\$title => alignment title
\$alignment => either fast or full
\$ktup => word size
\$window => window length
\$score => score type
\$topdiag => top diag
\$pairgap => pair gap
\$matrix => matrix (blosum, gonnet, pam, etc)
\$gapopen => gap open
\$endgaps => end gaps
\$gapext => gap extension
\$gapdist => gap distances
\$output => output format
\$outorder => output order
\$tree => tree type
\$kimura => correct distance (on/off)
\$tossgaps => ignore gaps (on/off)

The sequence(s) can be passed in a text area or via a file that will be uploaded from the user to the server. Also, there is the possibility of specifying an email address so that the analysis output can be sent to the specified email address once the data is processed.

➤ **The wclustalw.php script:**

This script is a wrapper to the official Clustal W utility so that it can run through a web server. It takes the different options set by the user in the index.php script, parses them and calls the Clustal W and the njplot programs to process the input data. The output will be in HTML format if received by a web browser or in text format if received by email.

➤ **CTest and RunClusAnI:**

These are two shell scripts that setup the environment just before calling the Clustal W program. They are called by the wclustalw.php script.

All the PHP scripts are [Shibboleth](#) enabled, so only authenticated users are able to access them. For more details about these scripts and installation steps please consult the `./Biotools/Clustal/README` file on the tarball.

❖ **MPIBlast Code**

After installing all the programs in our system, we can start creating a web interface for mpiblast. We created some PHP scripts that wraps around mpiblast and creates a nice interface for web users. A brief description of these scripts can be found below. For more technical detail please look at the source code provided with this report. The code is fairly readable with useful comments.



➤ **The index.php script:**

This is the index page for our web interface. The web form has been kept simple. It takes sequence data input as copy/paste in a text box or as a sequence file upload or a GenBank accession number. It also asks for which database to blast against and how many results and alignments the user wants. Once submitted the file calls RunBlast.php to process the user input.

➤ **The RunBlast.php script:**

It parses the user input and generates couple of temporary file to run MPI Blast on them. Three temporary files get generated; one to hold the data in FASTA format, one to store output and one to write a script that actually invokes MPI Blast and uses the other files. Once finished writing the script submits the temporary MPI Blast execution script to PBS which can take advantage of a distributed computing environment. In an absence of PBS one can also try executing the shell script directly from the PHP script and just wait for it to finish. After submitting the job the page waits for 3 seconds and redirects to the result page. The reason for 3 seconds wait is to provide some time for small jobs to finish before we go to the result page.

➤ **The viewResult.php script:**

This script gets the output file name and checks whether the file has already be generated by MPI Blast or not. If it exists then it displays the entire content otherwise it prints out an URL and tells the user to comeback to that address later. It also refreshes itself every 30 seconds to check if the result has been generated.

You need to modify all three of these files to make it appropriate for system. The changes are mostly change of path to temporary file directory and installed software directory.

❖ **WWWBlast Code**

Once we have the standard package running properly we can start replacing the files to customize it. We will use some PHP scripts that will modify the way the front end interacts with underlying utilities and we will also make some changes to the output. A brief description of these scripts can be found below. For more technical detail please look at the source code provided with this report. The code is fairly readable with useful comments.

➤ **The index.php script:**

This is our new index page for WWW Blast replacing the old one. It servers the same purpose as collecting user inputs but calls a different script to generate the result. The web form incorporates all original inputs and calls callWWWblast.php for further processing.

➤ **The callWWWblast.php script:**

It parses out all the user inputs and calls the original blast.cgi program to generate result. But instead of displaying the result right away, it stores the entire output in a variable for further processing.

As a part of the result page there is a block of information that contains the name of the sequences that matched against the user input together with a link to NCBI's sequence view web site. Since some of our databases are not recognized by NCBI,



Tigr for example, these links will not take users to right place. So we use a home grown utility named ParseBlast to replace these links with the correct links to the Tigr web site.

To use this utility once all the output including the alignment layout image has been generated, the script takes out a part of the output file starting from the line "Sequences producing significant alignments" to the first occurrence of "</PRE>" tag. This part of the output needs to be modified. The script writes this chunk of output in a temporary file, runs the ParseBlast program on it and gets the output back. The output is an array of right links for those sequences. After getting this output, the script does a global replace of old links with new links to fix this issue. After this modification the script ends with generating the final modified result page.

