Thebes and the Concepts of Middleware

Connecting the various players in any inter-process communications require that the processes be able to discover each other, can authenticate themselves, and can evaluate authentication to make authorization decisions based upon policies. This is the glue that connects resources, authentication, authorization, and discovery. Solving these problems in a scalable manner allows the creation of broadly distributed trustworthy systems capable of sharing any sort of resources.

However, scalability has been lacking to date. The Thebes project was created initially to solve these problems in the grid computing arena. Grid computing died primarily because no one was able to accomplish scalable solutions to these problems. Now the magic word is cloud computing. Unfortunately, the use cases that brought about interest in grids still remain unsolved.

Now, starting out with generous funding from Sun Microsystems, Thebes has expanded in scope to include any place where middleware is relevant. Feel free to explore the core technologies that Thebes is using, then explore the books regarding various arenas where these technologies apply.

Philosophically, the Thebes Consortium believes the following statements are true and must be considered when designing a new grid infrastructure:

* With some exceptions, resource owners generally do not care about the identities of the user.
* Resource owners should not require advanced, or pre-negotiated knowledge of remote users.
* Every resource must be able to enforce policy.
* Resource owners must be protected from execution of dangerous applications.
* Every resource must be able to track usage.
* Users prefer there be no distinction between local and grid resources in their ease of use.
* Users seldom care where their job is executing.
* Users should have automatic access to all resources whose policies they satisfy.

This site has been reorganized to reflect expansions in goals and a new site layout. If there is something you are looking for that is not yet restored, use the contact page to request it.

Finally, if you have a new book you want to enter, use the Contact page to send an email requesting an account.

Arnie Miles
01/12/09

The Thebes Consortium Project: A Call to Arms!

It is difficult to combine (federate) resources.

Current grid middleware has not caught the interest of anyone outside the major, nationally funded grid projects. Smaller and corporate enterprises are attracted to the ideas, but do not install the available software.

The Thebes Consortium is a group of corporate, higher education, private, and government entities that feel that the concept of grid computing is a valid and important one. We believe the implementation so far have failed because they have not provided scalability, security, ease of use, and have not attracted a community of users. We propose to rectify the situation as a consortium.

This site started as a whitepaper, and now is being used to throw open the gates to the community to help build these concepts and start writing a new breed of grid middleware. Please create an account. Individuals can add comments to anything on the site. Members of the consortium can write new content, and we are now soliciting use cases.

How can you participate? As an individual, you can read and comment on what's written on this site. If you are willing to commit as a member, you will be able to create new content on this site, and will be able to start contributing use cases immediately and code when the time comes. we need:

- Money to fund developers, graduate students, equipment, and travel at GU and other institutions.
- Dedicated time from any party interested in creating use cases, developing, alpha or beta testing, and commitments to installations of final code.

Contact Arnie Miles from the Contact page to become a member to start commenting on our ideas.

Security Token Service

The Security Token Service was started by and owned by SWITCH, which is the company supporting the Swiss Grid. Once completed, the STS will accept any of a number of security tokens that a user may possess and return the token the user needs.

This Authentication "rosetta stone" is waiting for a qualified Java developer with experience to help us pick up where it left off and finish under SWITCH's supervision and oversight. This component is critical to Thebes and will be a major component in security mechanisms in the foreseeable future.

Interfacing with a resource

The Security Token Service will, when complete, output whichever token a user needs to interact with a resource. For new development, Thebes is currently interested primarily in using the OASIS Security Assertion Markup Language (SAML) to create client interfaces, because SAML provides a rich set of detailed attributes about a user to be used in authentication decisions.

A Thebes Service Interface looks for SAML code, strips it and consumes it. Thebes released a demonstration TSI bundled with a Distributed Resource Management Application API (DRMAA) interface to Sun Grid Engine. This bundle accepted SAML assertions and compared the results to a set of course grained policies to make authentication decisions. Work is continuing to create a production ready example of this that will work with any job scheduler.

This is just one example, however. TSIs need to be created to satisfy a broad spectrum of use cases. This book will discuss current progress in this arena.

- Interface with a Computational Element
- Interface with a Data Element
- The Interface

Interface with a Computational Element

The first effort by Thebes was a proof of concept project called Condor-Shib, written by Brent Putman and members of the University of Wisconsin Condor team.

Georgetown University graduate school student David Cafaro created the demonstration instance of the Thebes Service Provider bundle for DRMAA. This was demonstrated at the 2008 Summer Sun HPC Consortium meeting.

Tim Bornholtz, as an employee of the middleware team at Georgetown University has taken this project and is in the process of creating a production ready interface for job schedulers. His test case is Sun Grid Engine, but since he is using JSDL as an input and DRMAA as his output, this client should be readily adaptable to other job schedulers. This project is very modular, and the plan is to reuse some of this code in the file system and sensor access mechanisms.

- The Condor Shibboleth Integration Project
- Initial Prototype of a Resource Interface

The Interface

Command Line
There is anecdotal evidence that some users still prefer to interact with their systems via the command line. As the goal of this project is to build a new system of grid middleware based upon use cases that are collected, we must allow for the possibility that a command line interface to the grid is still desired.

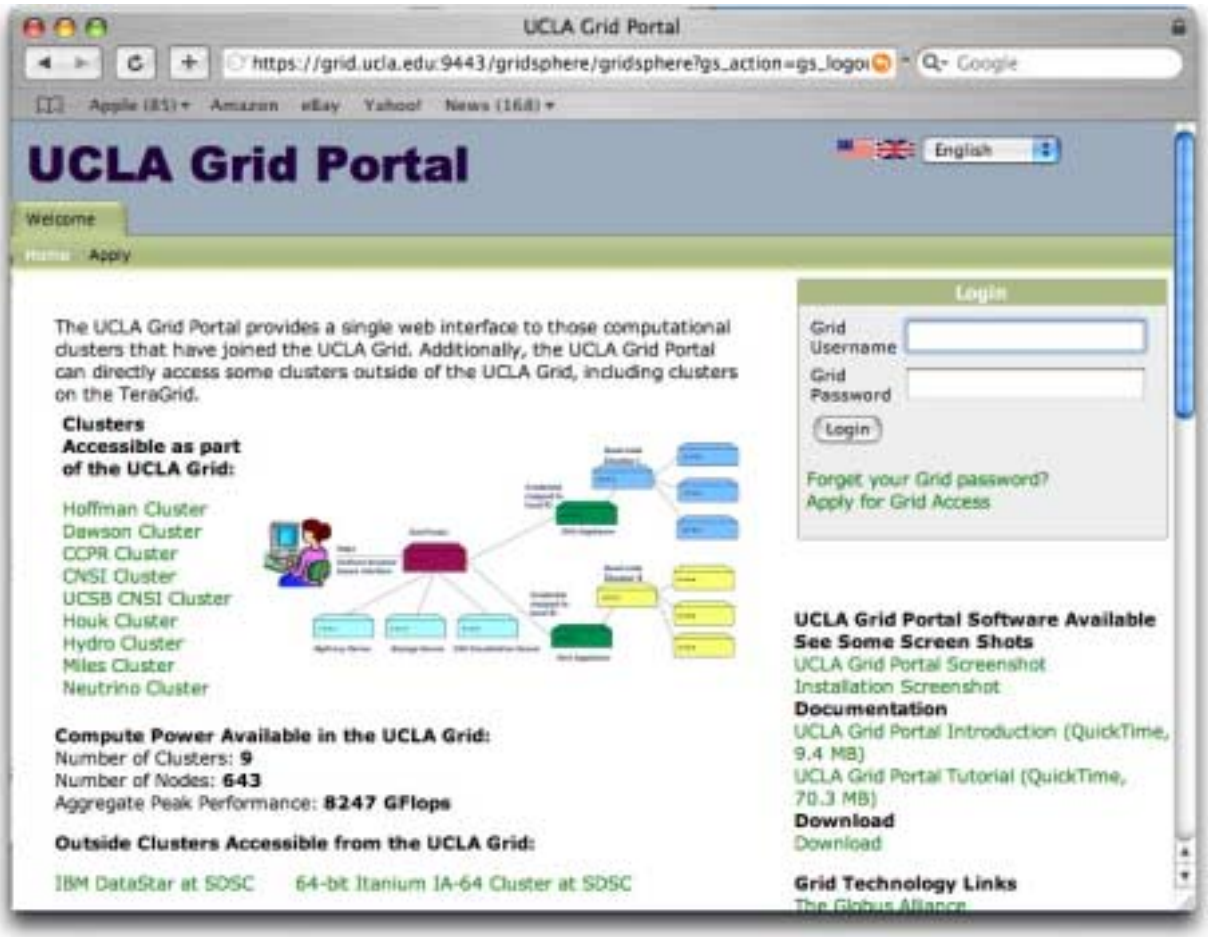
A command line solution must still provide protection from the execution of rogue code.

Graphical User Interface

The project should be prepared for use cases that require a locally installed client.

Portal

Providing users with a portal by which they access the grid brings numerous advantages to the project. There is considerable work being doing in this arena, one example is the UCLA Grid Portal depicted in the graphic.



UCLA GRID PORTAL

The portal will not only provide users with a common interface to access the grid, it will also serve as a aggregation point that can display available applications, licenses, resources, data storage locations, sensors and any other resource on the grid. The portal would only display those items available to the user based upon the attributes provided at sign-on.

Interface with a Data Element

This is a placeholder. Discussions are ongoing with a University that is interested in working on an Interface with a Data Element.

Resource Discovery

Resource discovery requires a method for describing resources and a distributed method for advertising these resources.

- **Resource Description Language**
- **Resource Discovery Network**

Resource Description Language

Tim Bornholtz has published the first draft of the new Resource Description Language data model, which is posted on https://svn.middleware.georgetown.edu/thebes-schema/. This is open for discussion right now, please take advant

Resource Discovery Network

Current grid discovery services are not actually focused on discovery, instead they act as a sort of grid DNS that resolves service identifiers into communicable endpoints. However, this functionality is not sufficient for the level of usage individuals have come to expect from Internet related systems; a search engine service for grids is necessary. Such a service will need to move away from the centralized, logically singular, index service with loosely, predefined, service metadata to a decentralized web of indexes that contain a much more verbose semantic description of a service.

Once a language is created to describe resources, a network of nodes arranged in a combined hierarchical and peer-to-peer arrangement will provide a robust method for user discovery of existing resources that both match a user's needs and authorization criteria. Thebes calls this the Resource Discovery Network.

- **Discovery**
- **OGF Draft OGSA Resource Selection Service Specification**

Use Cases

Thebes was founded on some core grid use cases that have existed since the dawn of shared resources. These should be documented here. We also solicit use cases from external users and potential users.

If you have an account, feel free to add pages to this book. If you do not, use the Contact link to request on, or submit a use case to be written up on your behalf.

- **Adding a Resource to a Confederation**
- **Distributed Attribute-based Authorization Service (DAAS)**
- **Submitting work for execution on a grid**

Adding a Resource to a Confederation

Goal in Context: A resource owner (desktop, HPC device, cluster, cycle scavenging campus grid, etc) wishes to share the new resource with all or some subset of the users allowed by the owner's enterprise by virtue of the confederations the enterprise belongs to.

Scope: Resource Owner

Level: Primary Task

Preconditions: The enterprise which the resource owner belongs to is a member of one or more confederations. High level policies have been agreed to regarding use of resources at the enterprise level. Other policies might exist at other levels, such as department level or division levels, that are more restrictive then the enterprise policies, that the resource owner may or may not be required to comply with.

Success End Condition: The resource owner will connect a resource as defined above to the confederation, using all policies the owner is required to comply with in addition to whatever policies the resource owner may wish to impose. The resource will have one of a variety of interfaces with the confederation, such as the job scheduler of choice, license manager of choice, etc. This interface will be plugged into one or more Resource Discovery Nodes, to which it will advertise its availability via a defined resource discover language. The resource will become immediately available for use by select users who belong to the confederation based upon the combined policies defined by the resource owner.

Failed End Condition: The owner is unable to publish a resource in such a manner that users can utilize it. The owner is unable to enforce policies.

Primary Actor: Resource owners, enterprise management, users

Trigger: The successful installation of a resource in an enterprise that needs to be connected to the confederation.

Main Success Scenario:
1. A resource is installed
2. The software that will be interfaced with the confederation is installed (job scheduler, file system, license manager, application, etc)
3. The confederation interface client software (Thebes) is installed
4. A hierarchical policy statement is created.
5. A resource description is created in a standard resource description language.
6. The resource is published to the nearest node of the resource discovery network
7. The resource becomes available for consumption by users in the confederation based upon the resource policies.

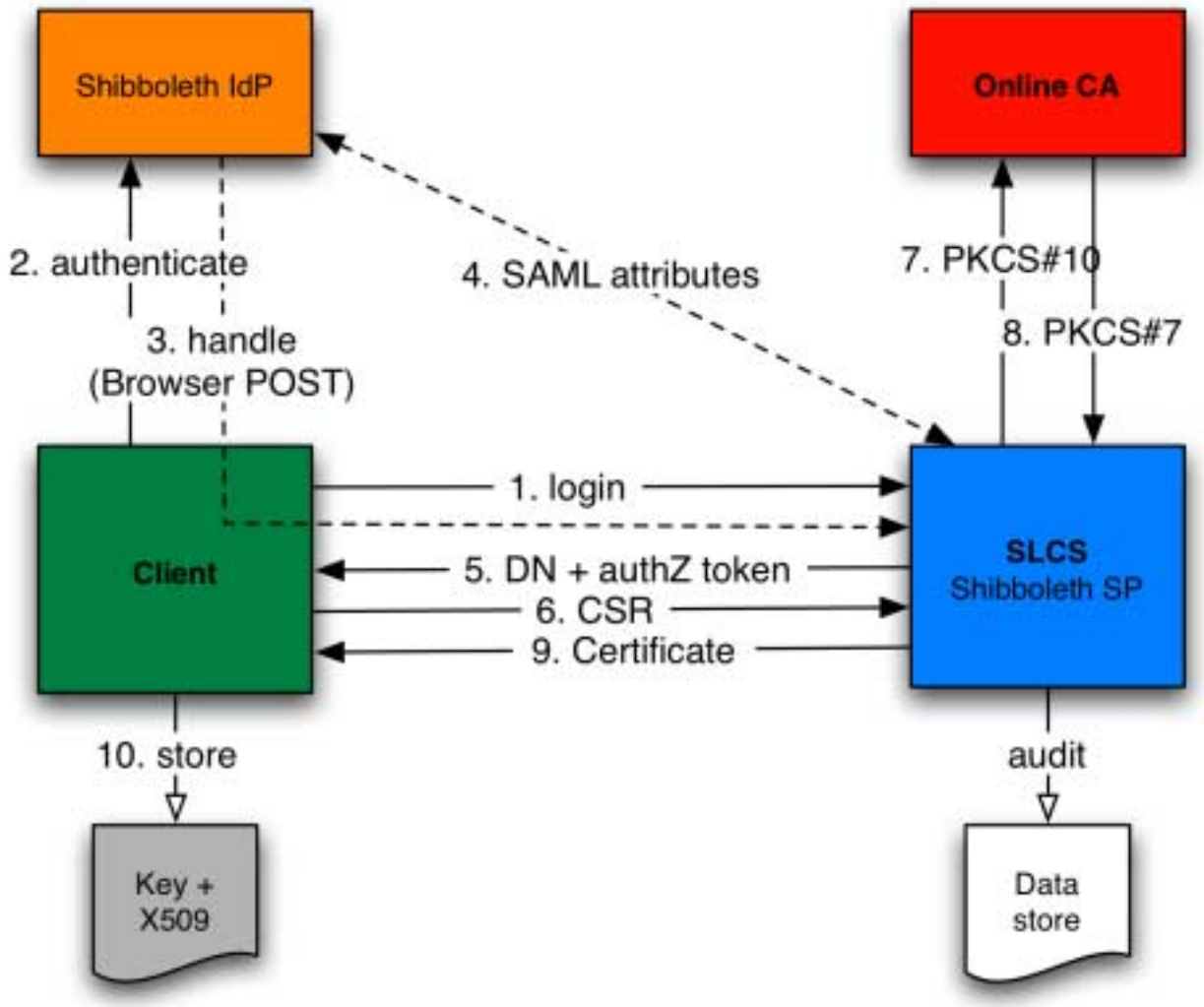
Distributed Attribute-based Authorization Service (DAAS)

In order to have all users in University of California campus to authenticate and authorize the use of HPC resources using Shibboleth, we will need to implement a distributed attribute-based authorization service. The following is a simple goal that we would like to achieve:

1. A user authenticates himself/herself with UC campus Shibboleth Identity Provider.
2. IdP returns to SP with some attributes which uniquely define that user. (say: ePPN)
3. Request a certificate from SLCS server which returns us a short live certificate
4. Use that Short lived certificate generate a proxy certificate for accessing grid resources.
5. We use that the proxy certificate to make a request to DAAS (which are distributed across UC)
6. DAAS returns many attributes for that user associated with that resource.
7. once we know that user authorization attributes, we know whether
the user can access the resource or not. or any special policy or any

allocation for a particular cluster and user.
8. The DAAS must be distributed in order to be scalable. That means each campus must have that DAAS deployed.
9. We will have DAAS on campus level as well as in UC level.

The diagram is like the following: (That one does not have DAAS)



We want that DAAS defines the user authorization attribute in a very simplest way such as:

whether they can access that resource or not?
Resource can be: storage, cluster, job service, pool service,etc

Finally it must also allow us to trace back who did what. (Audit needed)

We want to do it with a secure method. No middle man attack please.

Submitting work for execution on a grid

Goal in Context: A user submits a job to 'the grid' that requires compute, storage, license, application and/or sensor resources, and receives a response back when the work described by that job submission is completed, and where to find the results.

Scope: User

Level: Primary Task

Preconditions: One or more organizations have established a relationship with each other that allows the sharing of resources. User has a computational job to run. Data, application, licenses are identified. Physical resources are unknown. The user has access to one or more clients (CLI, GUI, Web) to submit work. One or more resources exist in the grid that the user has rights to use.

Success End Condition: The job is automatically directed to resources that meet the user's needs that the user is authorized to use, and the job is executed. The results of the job are places somewhere the user can access them, and the user is notified of the location and how to retrieve the results.

Failed End Condition: At any point in the process, the user cannot access a resource that is needed, causing a failure in the completion of the job.

Primary Actor: Any user

Trigger: User launching a job on the grid.

Main Success Scenario:
1. User creates a batch submission file describing all the details of the job to be run, including the location of those things the user is aware of (input data, applications, licenses, etc) and describing what other resources should look like (computational nodes, processors, memory, OS, etc).
2. User logs into whatever mechanism is used by the local enterprise, and attaches attributes to the batch submission file via some mechanism.
3. User submits batch file with attribute information to the nearest grid discovery node.
4. The grid has a mechanism to locate the closest resources that meet the user's needs.
5. The grid transports the work to the resources.
6. The resources verify the attributes about the user.
7. Resources perform work.
8. Results are written.
9. User is notified upon completion with instructions regarding collecting results.