AUTHENTICATION AND ACCESS CONTROL IN PORTALS: THE PROGRESS GRID ACCESS ENVIRONMENT

Michał Kosiedowski[1], Paweł Słowikowski[2]

[1]*Poznań Supercomputing and Networking Center, ul. Noskowskiego 10, 61-704 Poznań, Poland, kat@man.poznan.pl*
[2]*Academic Supercomputing Center of AGH University of Science and Technology, ul. Nawojki11, 30-950 Kraków, Poland, ps@agh.edu.pl*

Abstract

Web portals serve as an entry point to various network based services and allow for personalized access to resources. They must be able to identify the user in order to deliver the personalization feature. Further more, the portals must be assured that the identified user is allowed to utilize the resources. In the paper we discuss the ways to authenticate the portal user and to store the portal users database. We also take a look at the Resource Access Decision model as a possibility to deliver authorization services. Finally, we present the solution proposed within the PROGRESS project by the joint team from Poznań Supercomputing and Networking Center, Poznań and the Academic Supercomputing Center of AGH University of Science and Technology, Kraków. The PROGRESS grid access environment, with its distributed architecture based on Web Services communication, set challenging requirements for an authentication and authorization scheme. We solved the problem of authentication and authorization between distributed PROGRESS modules by applying a system consisting of the portal framework authentication, the RAD based authorization and the identity server's Single Sign-On mechanism.

1 Introduction

Web portals are widely known to serve as an entry point to various network based services. Web systems like Yahoo.com [1], onet.pl [2] or wp.pl [3] are used as the front-end to the Internet. Informational portals like cnn.com [4] or gazeta.pl [5] deliver news. There are also multiple specialized portals like governmental, such as the Multimedia City Guide [6], or educational, such as the Polish Educational Portal [7, 8], ones. Recently, e-science portals which provide computational services and serve as grid access environments have emerged; examples include HotPage [9], the Information Power Grid portal [10] and the PROGRESS HPC Portal [11, 12]. All these portals have one feature in common: they intend to deliver personalized workplace.

Delivering personalization functionality requires portal frameworks to be capable of identifying its users. User identification, together with a mechanism controlling access to portal resources allows to deliver services adequately and securely. In this paper we present the solution chosen and implemented for the authentication and authorization of users in the

PROGRESS grid access environment. First, we introduce the available solutions for user authentication and resource access control. Then we show how the Resource Access Decision specification can be used as the basis of an authorization system for portal services. Finally, we describe the authentication and authorization model designed for the PROGRESS HPC Portal.

2 Authentication and authorization mechanisms in portals

Portals, like any other application, need to be equipped with a mechanism to authenticate users for proper identification of users. The most widely used technique of authenticating users is confronting the `username+password` pair provided by the user with a database. The user fills in the fields in a form provided on the portal's web page. The browser passes the data to the portal in a request of login. The login portlet, which is responsible for the authentication of users, checks whether the username exists in the user database associated with the portal and, if it does exist, checks whether the supplied password is correct. The portal often offers a possibility to store the username and the password on the user's browser file in the form of a cookie. The cookie is then automatically passed to the portal whenever the user accesses the portal's pages. The other technique which can be used for identification of portal users are certificates. The concept bases on the public key cryptography mechanism and is not only used to authenticate users, but also to encrypt the data transmission.

Despite the method used for user authentication the credentials supplied by the user must be confronted with the user database. Usually the user database contains information like user full name or e-mail address additionally to credentials. The information may be stored using directory services based on LDAP, a relational database system (such a solution is used for example in the Joshua portal framework[13]) or operating system user models. The successful confrontation of user's credentials with the user database leads to opening a session on the portal, which gets destroyed after the user logs off or the expiration time passes.

Authentication secures proper identification of users. However, it does not assure the portal that the user can access a particular resource delivered by the portal. Therefore, portals must use an authorization mechanism which enables authorization of users' requests. The authorization mechanism should be capable of giving an answer of "YES" (the user can access the resource) or "NO" (the user cannot access the resource) basing on the authorization database where users rights are stored and the policy assigned by the portal operator. The authorization database can again be based on an LDAP solution or a relational database system; the latter is used for example in the Joshua framework.

3 Resource Access Decision authorization model

An interesting solution for access control system for portal accessible services is basing the authorization mechanism on the Resource Access Decision (RAD) [14] model created by the Object Management Group's (OMG) Healthcare Domain Task Force (HDTF). RAD is a security service that allows obtaining authorization decisions and administrating access decision policies. Making an access decision in RAD depends on the resource to be accessed, the operation on the resource and credentials of the request's caller. RAD is designed for security-aware applications, which means that the code obtaining an authorization decision from RAD has to be added to an application explicitly. The main objectives of RAD are: decoupling the authorization logic from the application logic, allowing application of different authorization policies, and maximum simplification of secure application development based on RAD. RAD is a very flexible and extensible architecture. It can support complex and dynamic authorization policies, decentralized and heterogeneous authorization databases, and the securing of different types of resources.

Figure 1 shows the process of assessing an access decision when the client application's user (caller) requests access to a resource.

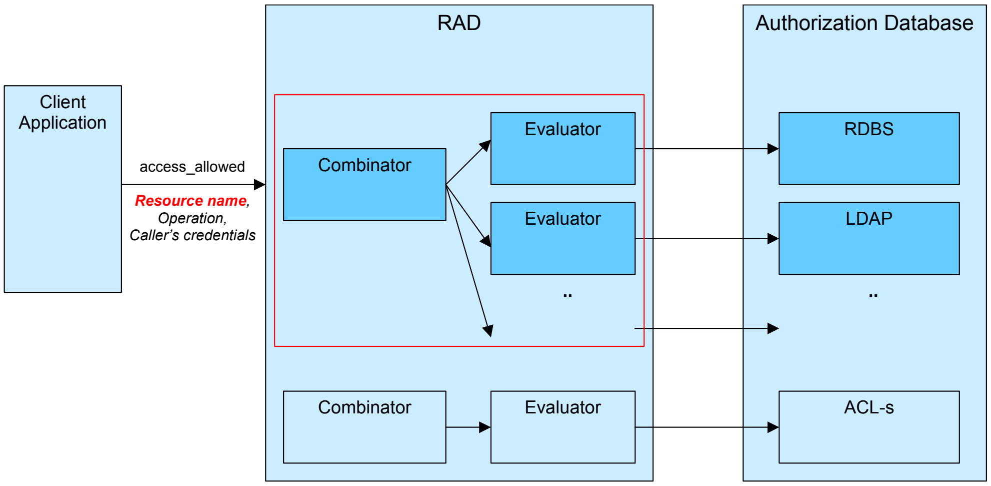

Figure 1 Access decision in the RAD authorization model

1. Whenever it is required, the client application checks whether the caller has granted access to the requested resource. In order to do so, the client application invokes the `access_allowed` method of the RAD interface passing three arguments: resource name, operation to be executed on resource and callers' credentials.
2. RAD selects evaluators and a combinator based on the resource name. Evaluators are responsible for interpreting authorization policy that controls access to the requested resource. In order to evaluate access request the evaluators refer to the authorization database. The combinator is responsible for combining the results from evaluators according to the authorization policy. The result of the combinator, a logic value granting or denying access to the resource, is returned to the client application.
3. Client application grants or denies access to the resource on the basis of the result received from RAD.

The PROGRESS grid services, including those of the grid service provider and the data management system, are security-aware applications. It is easy to define points of authorization requirements within them. This and other features of the RAD model, like support for complex and dynamic authorization policies, securing different types of resources and full support for a distributed environment contributed to its selection as the access control architecture for PROGRESS. In the next section we show how this authorization model has been incorporated into the PROGRESS grid access environment.

4 Authentication and Authorization Scheme in the PROGRESS Grid Access Environment

4.1 PROGRESS grid access environment

Figure 2 presents the architecture of the PROGRESS grid-portal environment. The PROGRESS computing grid consists of a distributed cluster of Sun computers connected via the PIONIER national academic network. It is managed by a combination of Globus and Sun Grid Engine software and accessible with the use of the grid resource broker [15]. The scientific data used for computing experiments performed in the grid is stored within the data management system [16]. The data management system serves as the source of input data and the destination of results for the grid jobs. The user may utilize the grid and data resources using the PROGRESS grid access environment. This environment, the HPC Window [17], consists of user interfaces: the web portal, which is the main user interface, and the migrating desktop [18], and the grid service provider [19] which is the middleware layer between the user and the computing resources.

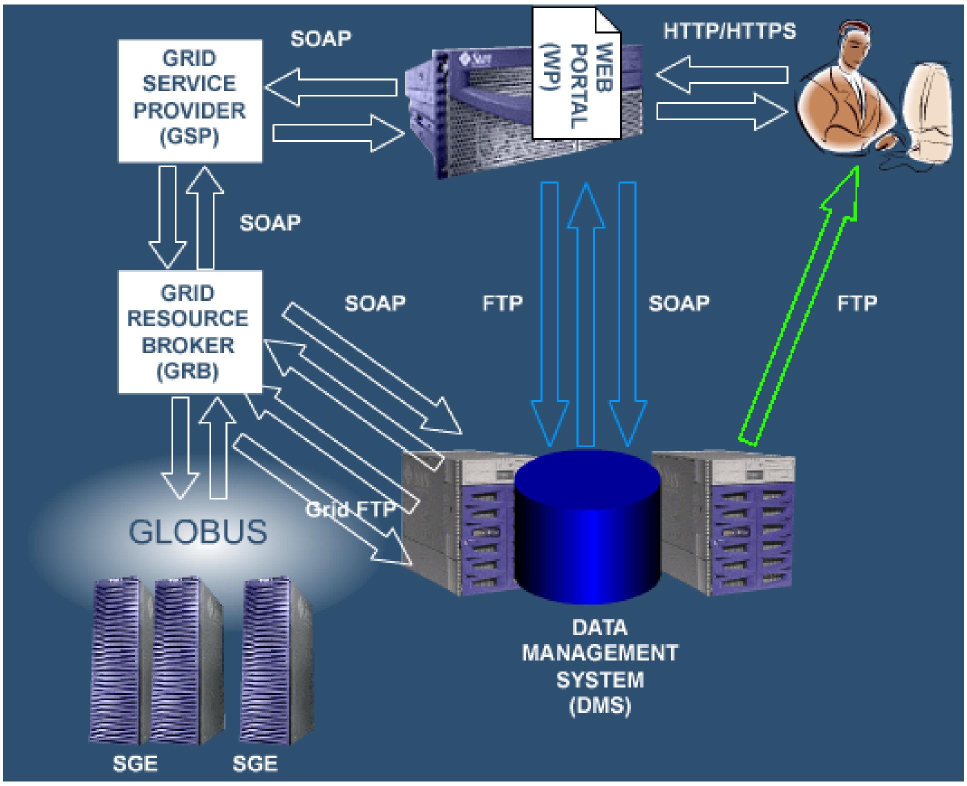
GRID SERVICE PROVIDER (GSP)
SOAP
WEB PORTAL (WP)
HTTP/HTTPS
SOAP
GRID RESOURCE BROKER (GRB)
SOAP
FTP
SOAP
FTP
Grid FTP
GLOBUS
DATA MANAGEMENT SYSTEM (DMS)
SGE
SGE

Figure 2 The PROGRESS grid-portal environment

4.2 Authentication model

Authentication in the PROGRESS grid access environment is based on the `user+password` technique. The user enters his/her username and password in a form presented to him at the portal's webpage (or in the migrating desktop). The data is then passed by the browser to the portal. The login portlet confronts the supplied data with the LDAP based user database. The process is performed with the mediation of the identity server.

The identity server has two responsibilities in PROGRESS: authenticating users (with the use of an LDAP database) and user session management. The latter function is utilized within the Single Sign-On (SSO) model. Whenever the user is successfully authenticated the identity server creates a new session marking it with a special token. The token is passed to the user; typically, if the user accesses the PROGRESS grid via the web portal, the token is passed in the form of an http cookie. Each time the user sends a new request to the grid service provider the invocation of the provider's method includes the token as one of the provided method invocation parameters. The grid service provider validates the token on the identity server before performing any actions on behalf of the user. What is more, the token specifies which user owns the session marked with the supplied token; the grid service provider uses this information to identify the user.

4.3 RAD based authorization module

When the grid service provider identifies the user, it is aware that it is required to authorize the user to perform the requested operation. The answer to the question, whether the service can or cannot be allowed to process the user's request, is provided by the authorization system. The authorization system in the PROGRESS grid access environment is based on the RAD architecture. As it has already been mentioned, the PROGRESS grid services are security-aware and the PROGRESS grid access environment is fully distributed between numerous computers and sites. The features of the RAD model inspired us to choose this architecture for the implementation of the required authorization system.

As it was presented in Section 3 the service must invoke the RAD's `access_allowed` method to authorize the user's request. The parameters encapsulated in the invocation are: the resource name, the operation name and the caller's credentials. In PROGRESS we assigned this to:

- the resource name – there are currently three types of resources in PROGRESS;
- the required rights – the grid service being aware of the security can provide the authorization system with the required right for the operation performed by the user;
- the user's distinguished name as extracted from the token passed to the grid service provider by the method invoker.

Basing on the nature of services we defined three types of resources for the grid service provider. These types are:

- the service instance – this type is used for identifying an instance of a service which can provide multiple instances to users. The service instance concept was designed for distinguishing numerous logical spaces within a functional service module. It can be mainly used by informational services like the short news service (the currently implemented example of an instance-enabled service), the link directory service or the discussion forum service. These services enable to define multiple logical areas within them. For example, it is easy to imagine two short news instances: one for presenting news about bioinformatics and the second presenting news about the PROGRESS project. Both can use the same functionality of the service, but can have their own logical space within the service database. The service instance is identified by the pair `SERVICE_NAME, INSTANCE_NAME`.

- the application – this type is used for identifying applications available in the application factory managed by the application management service. An application is identified by the `APPLICATION_NAME`, which in fact is the application identifier within the factory.
- the computation – this type is used for identifying the grid jobs created within the job submission service. Each grid job is identified by the `COMPUTATION_NAME`, which in fact is the grid job identifier assigned by the job submission service.

For the defined types of resources we assumed the user roles (rights) presented in Table 1.

Table 1 Types of resources within the PROGRESS grid service provider

Role (right)	Description
Service instance	
READER	The READER can read data within the service instance (for example read news)
EDITOR	The EDITOR can additionally edit data within the instance, that is add new elements, modify and remove the existing.
ADMIN	The ADMIN can perform all operations available within the whole service, including adding, modifying and removing instances.
Application	
USER	The USER of an application can use this application to run computing experiments in the grid.
DEVELOPER	The DEVELOPER can add new applications to the application factory and can modify and remove applications that he/she authored (the application descriptor contains information on the owner of the application).
ADMIN	The ADMIN can manage the whole application factory: add new applications, modify and remove the existing ones.
Computation	
OWNER	The OWNER of a computation is the user who creates this job within the job submission service. He/she can modify the configuration of the job and delete it
READER	The READER of a computation is the user to whom the OWNER assigned rights to read the configuration of the job (and the experiment results assuming that proper rights are assigned within the data management system).

The assumed resource types are the types required by currently implemented grid services. There is a possibility to add new resource types (and associate proper rights with them) if a new grid service is deployed onto the grid service provider and requires so. The resource types can be added with the use of the authorization database administration application. The RAD administration module allows to manage rights, that is add, modify and remove rights (and manage available resource types). It is a J2EE based application with a web graphical interface. What is interesting, the authorization system administration application is also authorized with the use of the PROGRESS RAD module; it has its own space of users, types of resources and rights.

4.4 Authentication and authorization scheme in PROGRESS

All processes necessary to properly authenticate the user and authorize the user's request within the PROGRESS grid access environment are illustrated in Figure 3.

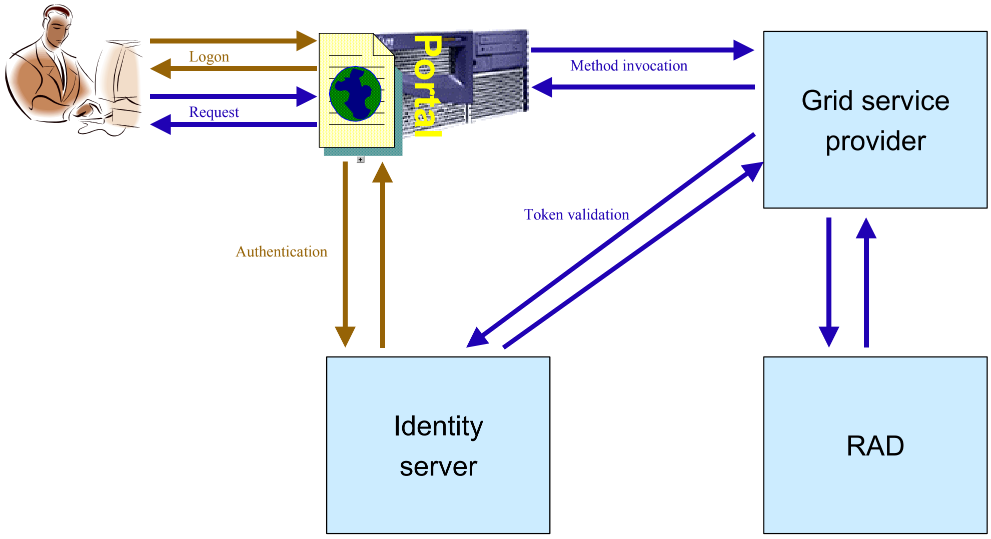

Figure 3 Authentication and authorization processes within the PROGRESS grid access environment

Whenever the user wishes to utilize a grid service he/she is required to log on to the user interface, for example the web portal. The portal contacts the identity server and authenticates the user with the supplied username and password. If the authentication succeeds, the identity server creates a new user session and marks it with a token. The token is placed by the portal within the user's browser environment as an http cookie. Next, each time the user requests access to a resource within the grid service provider the cookie with the token is transmitted to the portal. The portal encapsulates the token into the invocation of the required method of the grid service provider. The grid service provider validates the token on the identity server and extracts the username. Then, it requests the authorization system to decide if the user has the required right to access the required resource. Depending on the RAD's answer the operation is performed and the result data set is sent back to the portal and the user (if YES) or the grid service provider throws an exception indicating that the user does not have the right to perform the required operation on the resource (if NO).

5 Summary

Authentication and authorization play the key role in portals. In this paper we presented one of the possible ways to follow while designing the architecture for the security model. The PROGRESS HPC Portal, which provides access to grid and data services distributed along sites, set a freat challenge for us. A good authentication and authorization model was the key to the success of the distributed grid access environment. However, the RAD based authorization system, together with authentication and the SSO model based on the identity server, helped us to achieve our goals to create a comfortable work place and a safe grid access environment.

It is also worth mentioning that the authorization system deployed within PROGRESS is currently integrated with the grid service provider. All grid services within this module are authorized and incorporated into the SSO mechanism. The second module, which is to be authorized using the RAD based authorization system, is the data management system. The work on integration of these two modules is currently under way and will be finished in May 2003. The whole secure PROGRESS environment will be ready for official deployment at the end of that month.

References

1 Yahoo.com web portal. Accessed from http://www.yahoo.com/.

2 Onet.pl web portal. Accessed from http://www.onet.pl/.

3 Wirtualna Polska web portal. Accessed from http://www.wp.pl/.

4 CNN web portal. Accessed from http://www.cnn.com/.

5 Gazeta.pl web portal. Accessed from http://www.gazeta.pl/.

6 Multimedia City Guide. Accessed from http://www.city.poznan.pl/.

7 Polish Educational Portal. Accessed from http://www.interklasa.pl/.

8 C. Mazurek, P. Partyka: Polski Portal Edukacyjny. Proceedings of the „Polski Internet Optyczny: Technologie, Usługi i Aplikacje – PIONIER 2002" Conference, Poznań, Poland (2002).

9 NPACI HotPage Grid Computing Portal. Accessed at https://hotpage.npaci.edu/.

10 Information Power Grid. Accessed from http://www.ipg.nasa.gov/.

11 PROGRESS HPC Portal. Accessed from http://progress.psnc.pl/portal/.

12 M. Kosiedowski, C. Mazurek, M. Stroiński: PROGRESS – Access Environment to Computational Services Performed by Cluster of Sun Systems. Proceedings of the 2nd Cracow Grid Workshop, Kraków, Poland (2002), pp. 45-56.

13 M. Kosiedowski, C. Mazurek, S. Szuber: Narzędzia do zarządzania zawartością portali. Proceedings of the „Polski Internet Optyczny: Technologie, Usługi i Aplikacje – PIONIER 2001" Conference, Poznań, Poland (2001).

14 Resource Access Decision, Version 1.0. Accessed from http://www.omg.org/technology/documents/formal/resource_access_decision.htm.

15 K. Kurowski, J. Nabrzyski, J. Pukacki: User Preference Driven Multiobjective Resource Management in Grid Environments. Proceedings of CCGRID 2001 conference, Brisbane, Australia (2001).

16 P. Grzybowski, C. Mazurek, P. Spychała, M. Wolski: Data Management System for grid and portal services. Accessed from http://progress.psnc.pl/.

17 M. Kupczyk, R. Lichwała, N. Meyer, B. Palak, M. Płóciennik, P. Wolniewicz: Roaming Access and Migrating Desktop. Proceedings of The 2nd Cracow Grid Workshop, Kraków, Poland, December 2002, pp. 148-154.

18 M. Bogdański, M. Kosiedowski, C. Mazurek, and M. Wolniewicz: Facilitating access to grid resources with the use of the HPC Window. Submitted for presentation at The 9th International Conference on Parallel and Distributed Computing Euro-Par 2003, August 26th-29th 2003, Klagenfurt, Austria. Accessed from http://progress.psnc.pl/

19 M. Bogdański, M. Kosiedowski, C. Mazurek, and M. Wolniewicz: GRID SERVICE PROVIDER: How to improve flexibility of grid user interfaces?". Accepted for publication and oral presentation at The 3rd International Conference on Computational Science, June 2nd-4th 2003, St. Petersburg, Russia. Accessed from http://progress.psnc.pl/.