International Journal of Computer Science and Applications © 2009 Technomathematics Research Foundation
Vol.6 No.1, pp. 1-12

A CROSS - DOMAIN ROLE MAPPING AND AUTHORIZATION FRAMEWORK FOR RBAC IN GRID SYSTEMS

G GEETHAKUMARI
University of Hyderabad,
Hyderabad, Andhrapradesh, India
geethamaruvada@gmail.com

DR ATUL NEGI
University of Hyderabad,
Hyderabad, Andhrapradesh, India
atulcs@uohyd.ernet.in

DR V N SASTRY
Institute for Development and Research in Banking Technology
Hyderabad, Andhrapradesh, India
vnsastry@idrbt.ac.in

Highly computational resource sharing environments like grids pose major security issues. Secure interoperability has been a growing concern for such multi domain computing systems. Collaboration in such a diverse environment requires integration of all local policies to compose a global access control policy for controlling information and resource. Access control in such an environment is still an open problem. The much standardized Role Based Access Control (RBAC) is yet to be fully utilized in a multi domain grid environment like the Grids. Here, we present an architectural framework for adaptation and implementation of RBAC for grid access control. Our approach includes solutions for delegation and revocation in a single domain grid enterprise. The classical Role Based Access Control, though an effective access control standard, does not address the issue of resolving a local role into a global role. So, we also propose an architecture based on RBAC, which can establish role equivalence among the domains by mapping a local domain role to its equivalent global role. We use the approach of weighted ranking for the same. The final authorization decision is made based on the mapped global role ranking and also the resource access policies.

Keywords: RBAC; Grid Computing; Security; Cross-Domain Authorization.

1. Introduction

Grid computing is regarded as an emerging technology of immense potential in both industry and academia [6], [7]. A grid environment supports resource sharing with scalability and heterogeneity [8], [15]. Security is of prime concern while sharing data and computational resources in a grid. As a computing environment which supports resource sharing with scalability and heterogeneity, the security procedure in grids involves authentication and authorization [13], [19]. The issue of grid authentication has been a research interest for long. Scalable and secure authorization [9] mechanisms are yet to be incorporated into a grid environment.

RBAC, a well established standard [17], [18], for access control has a lot of limitations in a grid scenario. Our work proposes to solve this issue through a novel architecture. Grid security consists of different measures such as naming and authentication, trust and policy management, authorization [13] etc. A grid security procedure starts when a user initiates a resource access. Controlling the access to any resource primarily involves two phases namely authentication and authorization. Existing grid security mechanisms are based on authentication and do not lay emphasis on authorization. We address this problem of authorization in grid system environments. Mandatory authorization schemes were initially used to define access control policies [14], [16].

The evolution of RBAC as a reliable standard for single enterprises motivated researchers to think of ways in which it could be incorporated into grid environment. A grid is often viewed as a logical organization formed of multiple physical organizations or enterprises and hence the integration of RBAC into grid is only a logical extension of the standard RBAC implementation. For a grid system usually formed by multiple domains which are maintained by different companies, organizations or institutions, interoperability is a major issue. A role, which is the basic unit of access control as per the RBAC, signifies different meanings in different organizational contexts. Here we arrive at a mechanism by which we can map the role of one enterprise into its new semantics in another enterprise.

This paper has been organized as follows. Section 2 gives the related research and also the motivation for our work. In section 3, we show the proposed authorization architectural framework for a single domain grid enterprise as well as a cross-domain authorization architecture based on ranking of roles. In section 4, we give an insight into the implementation details with a few select snap shots. We also suggest the possible enhancements. Section 5 summarizes our work.

2. Motivation and Related Research

Though there has been considerable work in the area of grid security, the emphasis has been on authentication. Grid access control and authorization are still open research issues, which needs much attention. In this paper, we focus on the issue of interoperability between different domains when it comes to the issue of authorization.

James B.D.Joshi et.al [2] suggest an integer programming (IP) based approach for secure interoperation involving RBAC policies. But their work does not reflect the distinct characteristics and requirements of grid authorization. Another proposed approach is user-credential based role-mapping where by a user's credentials associated with the role form the basis for role-mapping. We believe this is a premature and non-standard way of mapping roles, as the fundamental unit of RBAC is a role itself and hence cannot use its associated credentials as the sole criteria for role mapping. Liang Chen et.al [10], have proposed an inter-domain role mapping technique based on the principle of least privilege. They suggest a minimal cardinality for a role across a domain to avoid misuse of access. This again, does not suit dynamic and heterogeneous environments like the grids.

Some of the existing grid authorization mechanisms are Permis, Akenti, Shibboleth, VOMS, CAS. Though Permis, Akenti and CAS introduce the concept of roles in a grid environment, they are not role-based implementations like RBAC. Also they lack the flexibility of RBAC and are static in nature.

GSI, [19] the security component of Globus middleware supports authorization by way of proxy credentials. The drawback of proxy-based authorization is that proxy has short life time. Also, proxies have a disadvantage that the end-user may be away or disconnected, so issuing the proxy credentials dynamically may not be possible. Also the private key bound to a proxy credential cannot be stored in an encrypted form as it has to be read by the processes to which the rights have been delegated without contact with the end-user. Though online proxies are now supported by globus, they need to be periodically issued. The absence of a standard role-mapping mechanism to address the grid authorization and access control issues, combined with the fact that the present form of RBAC for single enterprises cannot support grid access control motivated us to develop a new architecture which can truly reflect a multi-domain grid access environment.

Some of the authorization attempts in a grid environment which have significance to this work include:

CAS [11] - Primarily a community based Authorization service, it allows resource providers to specify course-grained access control policies in terms of communities as a whole, delegating fine-grained access control policy management to the community itself. The major drawback is lack of scalability and denial of the basic right of every node to decide its users

RB GACA [20] - A RBAC based grid access control architecture. Based on the RBAC standard, the work deals with specifying an architecture for access control in multi domain environments. But it does not suggest any cross authorization methods among different domains. The need to develop a scalable architecture [9] for fine grained access control in a multi domain grid environment motivated us to propose and implement new architectures for authorization as well as role mapping across different domains in a grid environment.

3. Grid Authorization and Access Control Framework

According to RBAC, role is the basis for access control. Delegation and revocation mechanisms also are based upon role. In our work, we present an architecture proposed for implementation for access control with delegation and revocation models for a single domain environment. The following components form the core of the architecture shown in Figure 1.

- XACML framework: It is designed as a separate service running on remote system. It accepts the requests from the multiple authorization modules of the Globus toolkit. It makes decisions about accessing to the particular resource
- Globus container: This contains all the services and accepts requests from the grid clients. It is also running on a separate system
- Database: It contains information about user-role, role-delegation etc. PEP remotely connects to the database
- LDAP server: It runs on a separate system and accepts request from the LDAP clients. It contains full details of users and resources in the organization. It also responds to the PEP
- Access control policies: Policies are specified using XACML [3]. These policies are placed on the same system where the XACML framework resides

Our implementation includes interfaces for Admin of a resource, Client and the Administrator of the domain which we have deployed with the Globus toolkit container.

- Client GUI: This is with the Grid client system or may be another system
- Grid client - the client who is accessing a resource
- Grid Node - The node whose resources are being accessed
- Authorization module - This module takes the request from the Node and sends back a deny/grant result
- Policy Enforcement Point (PEP) - This takes the user credentials and other details and queries for the resource access with the PDP in the XACML format and sends back a deny/grant result
- Policy Decision Point (PDP) - The XACML request sent to it is queried upon with the set of available policies to check for access. It then sends back a reply in the XACML format
- Administrator - Manages the whole grid scenario for that domain by adding/editing policies for resource access. It can add/edit users, their roles and delegation and revocation rules

All the users in the domain are assigned roles based on their responsibility in the domain. The Administrator of the domain defines access control policies for user-roles. A resource request from a user goes through a series of steps as shown in Figure 1. A user can delegate his roles to other roles based on the policies and revoke them later. This makes the access control policies flexible and dynamic.

3.1. *Authorization Architecture for Multi-Domain Environments*

A grid system usually consists of more than one domain in a hierarchical/nested fashion. Therefore, cross domain authorization is an essential factor for multi domain access control [5], [20], [12]. We propose an extended architecture in Figure 2. In this architecture, the Authorization servers in individual domains follow the

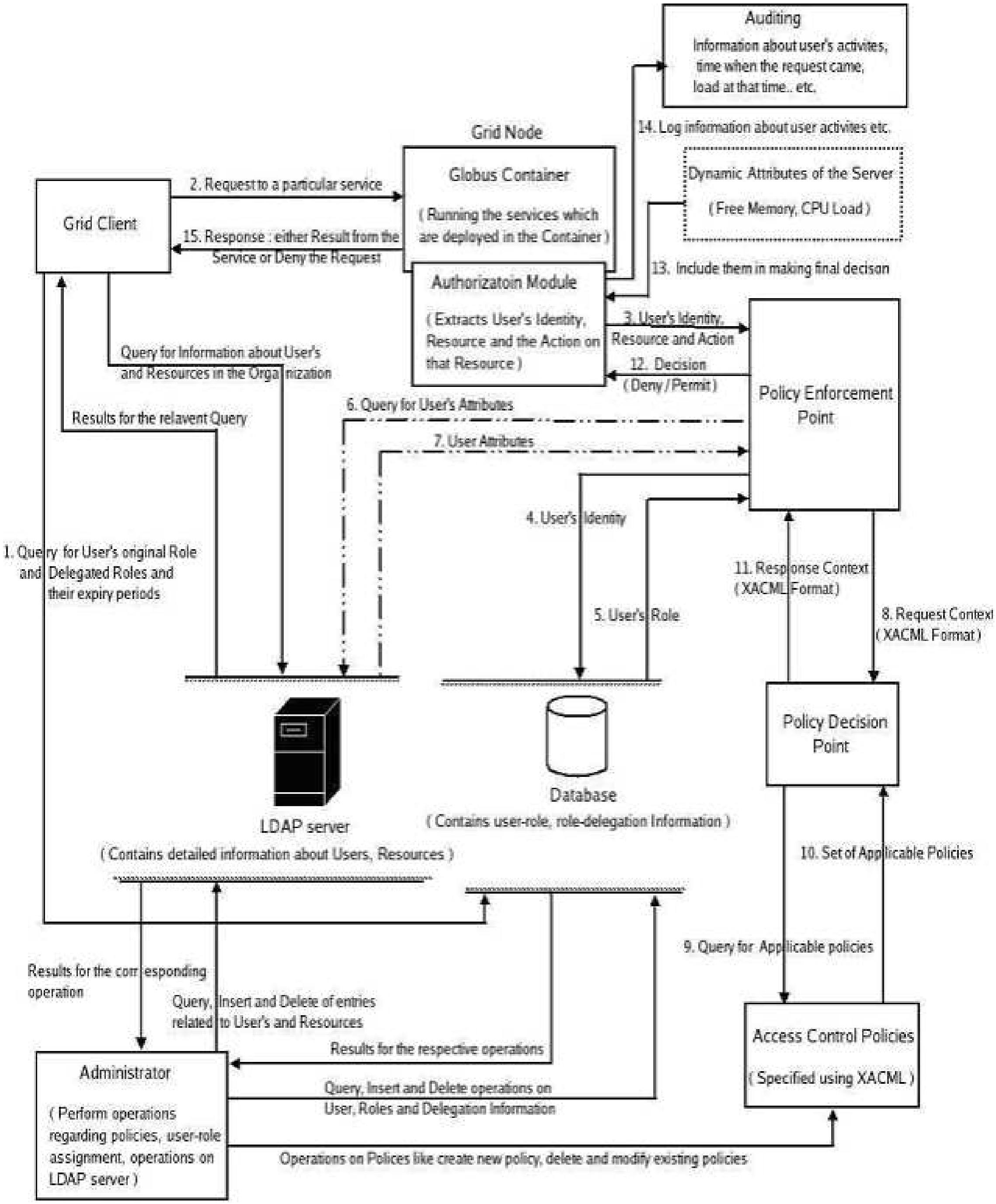
Auditing
Information about user's activites,
time when the request came,
load at that time.. etc.
14. Log information about user activites etc.
Grid Node
Globus Container
(Running the services which are deployed in the Container)
Dynamic Attributes of the Server
(Free Memory, CPU Load)
Grid Client
2. Request to a particular service
15. Response : either Result from the Service or Deny the Request
Authorizatoin Module
(Extracts User's Identity, Resource and the Action on that Resource)
13. Include them in making final decison
3. User's Identity, Resource and Action
12. Decision (Deny / Permit)
Policy Enforcement Point
Query for Information about User's and Resources in the Organization
Results for the relavent Query
6. Query for User's Attributes
7. User Attributes
4. User's Identity
5. User's Role
1. Query for User's original Role and Delegated Roles and their expiry periods
11. Response Context (XACML Format)
8. Request Context (XACML Format)
Policy Decision Point
LDAP server
(Contains detailed information about Users, Resources)
Database
(Contains user-role, role-delegation Information)
10. Set of Applicable Policies
9. Query for Applicable policies
Results for the corresponding operation
Query, Insert and Delete of entries related to User's and Resources
Results for the respective operations
Access Control Policies
(Specified using XACML)
Administrator
(Perform operations regarding policies, user-role assignment, operations on LDAP server)
Query, Insert and Delete operations on User, Roles and Delegation Information
Operations on Polices like create new policy, delete and modify existing policies

Fig. 1. Access Control System Architecture

architecture given in Figure 1. If the request is from a client in a different domain we map the roles using the concept of role ranking. According to RBAC, the roles in a domain are based upon a hierarchy. We make use of the hierarchical relationship to give roles a rank on a scale of 10. The request for the resource is passed on to the central authorization server which passes the request to the Authorization Server (AS1) of the domain in which the user is a part of. The AS1 retrieves the user's role and ranks it in its domain, creates a token and passes it on to central authorization server. It adds the rank for the domain and normalizes the rank of the client on a scale of 1 and passes the token to AS2. Here the subsequent rank of the source is added, normalized and compared with the rank of the client. If the required rank is greater, then access is denied, otherwise it is granted. If there are more sub domains then we find the normalized rank with respect to the first common ancestor between the client and the resource. If the user wants to delegate his role, then he passes on the produced token with the normalized roles. If the user who is being delegated is from a different domain, then the role normalization is done again with respect to the correct ancestor and a token is recreated.

The details of the user's delegated [12], [4] and revoked roles between two domains are stored with their common ancestor central server. The RBAC standard uses role as the basic unit of authorization. The standard incorporates features such as role hierarchy, static and dynamic separation of duties and so on. In an RBAC environment, a user will be assigned roles based on his responsibilities in the organization. For example, in a university domain, the potential roles could be Professor, Associate Professor and so on. For an industrial domain the roles could be CEO, General Manager, Manager and so on. Therefore, the semantics of roles in a given domain will not have relevance in another domain [1], [10]. Thus, the role in an organization has to be mapped to its corresponding meaning in another if cross-domain resource sharing is to be made possible.

We address this issue with a ranking based weighted role approach. Our architecture enables mapping of a local role to a global ranking. We consider a nested and hierarchical domain architecture reflecting the real life grid scenario. The roles in a particular domain follow a local role hierarchy. The cross domain architecture shown in Figure 2 consists of the following components.

- At the organizational level we consider two Domains A and B
- Domains A and B consist of sub-Domains A_ a and B_ b
- Further Domains A_ a and B_ b have grid nodes as their constituents
- Authorization Server1 (AS1) is the local Authorization server for grid nodes from Domain A_ a
- Authorization Server2 (AS2) plays a similar role in Domain B_ b
- Ranking servers RS1 and RS2 for the two respective Domains A and B store the rating of the subdomains

The whole grid environment is separated into different domains and sub domains as shown. The sub domains in every domain are given ranks on a scale of 10. The roles in a local domain are also ranked on the scale of 10 based on their hierarchy. The role in a local domain gets translated to a global ranking based on the value of its own ranking and also the rank of its ancestor domains up the tree. The role-mapping architecture is a weighted tree and we arrive at the globally mapped role by comparing the global rank of a role with respect to its first common ancestor as shown below.

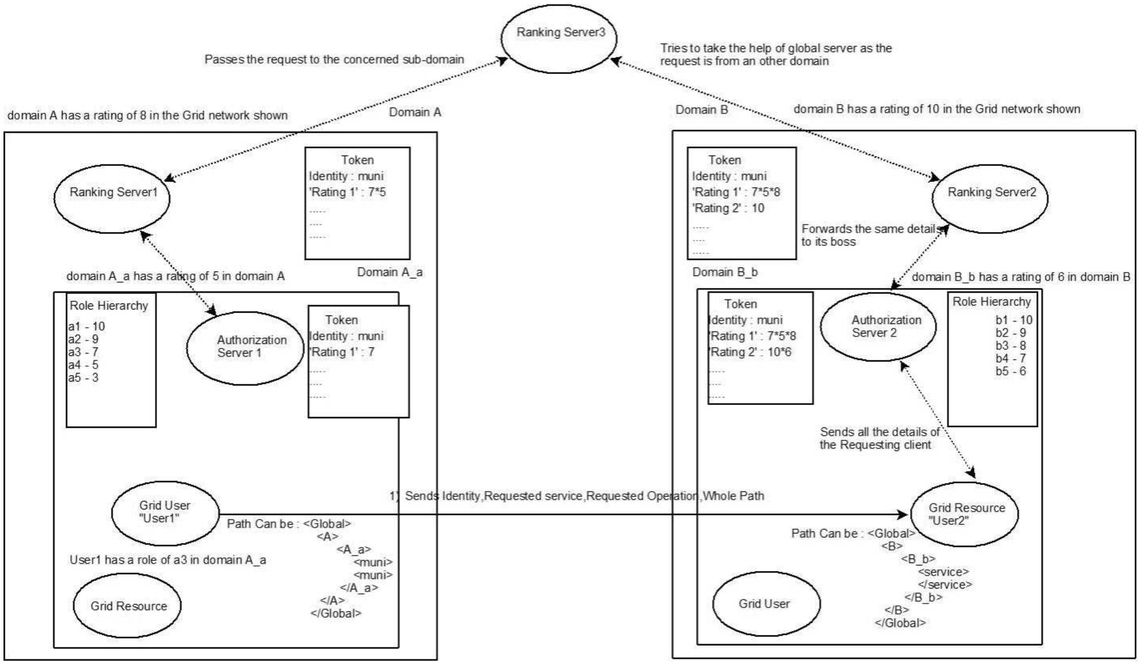
Ranking Server3
Passes the request to the concerned sub-domain
Tries to take the help of global server as the request is from an other domain
domain A has a rating of 8 in the Grid network shown
Domain A
Domain B
domain B has a rating of 10 in the Grid network shown
Ranking Server1
Token
Identity : muni
'Rating 1' : 7*5
Token
Identity : muni
'Rating 1' : 7*5*8
'Rating 2' : 10
Ranking Server2
Forwards the same details to its boss
domain A_a has a rating of 5 in domain A
Domain A_a
Domain B_b
domain B_b has a rating of 6 in domain B
Role Hierarchy
a1 - 10
a2 - 9
a3 - 7
a4 - 5
a5 - 3
Authorization Server 1
Token
Identity : muni
'Rating 1' : 7
Token
Identity : muni
'Rating 1' : 7*5*8
'Rating 2' : 10*6
Authorization Server 2
Role Hierarchy
b1 - 10
b2 - 9
b3 - 8
b4 - 7
b5 - 6
Sends all the details of the Requesting client
1) Sends Identity,Requested service,Requested Operation,Whole Path
Grid User "User1"
Path Can be : <Global> <A> <A_a> <muni> <muni> </A_a> </A> </Global>
User1 has a role of a3 in domain A_a
Grid Resource
Grid Resource "User2"
Path Can be : <Global> <B> <B_b> <service> </service> </B_b> </B> </Global>
Grid User

Fig. 2. Cross Domain Role Mapping Architecture

Figure 3 shows the user-role ratings for authorization.

The grid user is granted/denied access to the requested resource through the following procedure.

(1) The user from Domain A_ a sends his identity, path, the requested resource and also requested operation to Domain B_ b
(2) The user in domain b forwards the details to its Authorization Server (AS2) and awaits a deny or grant
(3) The Authorization Server AS2 executes the algorithm *Authorize* as shown above
(4) The credentials are passed up the hierarchy for role mapping as the user is from a different domain
(5) The credentials reach AS1 where the attributes like the user's role, rating etc

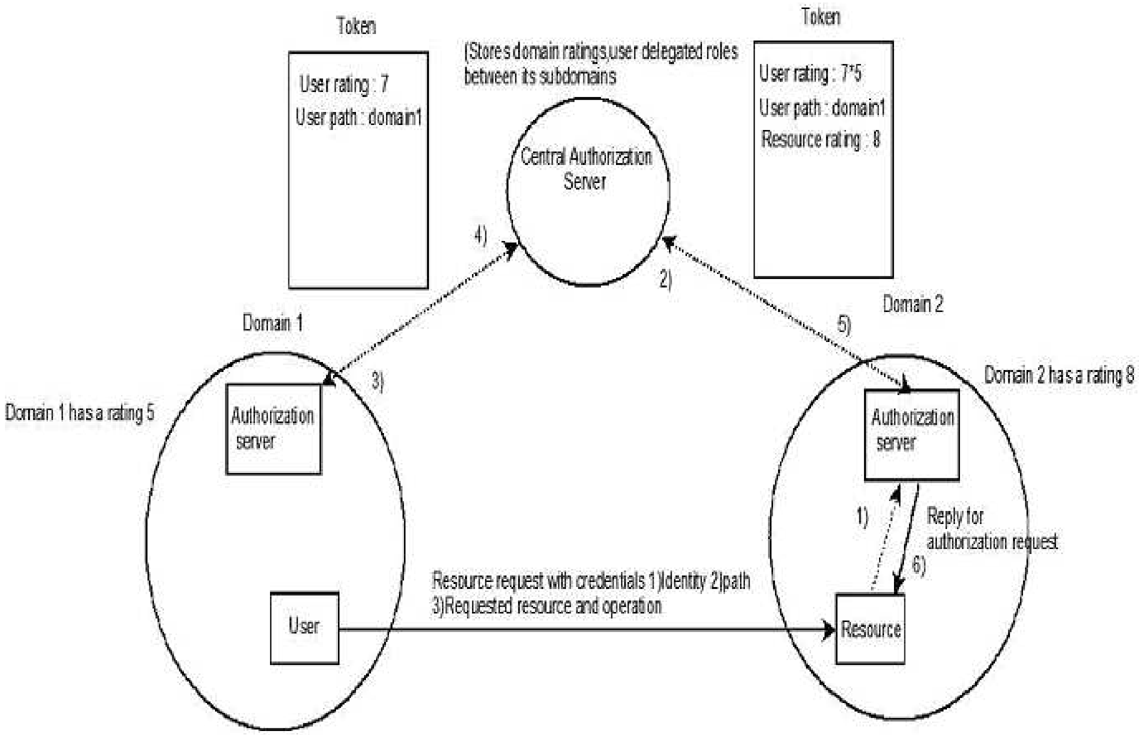

Fig. 3. User-Role Ratings for Authorization

are retrieved and a Token is created.

(6) The Token follows the same path in reverse and at every stage, the rating of the domain gets weighted
(7) AS2 gets the final version of the Token. It normalizes the user rating to 7*5*8 / 1000 = 0.280
(8) AS2 finds the minimal rating of a role needed to access the resource which is 5. So the Normalized rating of the resource is 10*8*5 / 1000 = 0.400
(9) AS2 can integrate this ranking comparison with other local policies to either deny or grant access to the user

Here, we give a generic algorithm for authorization procedure and role mapping.

```
Algorithm : Authorize
Input : User Credentials
Output : Grant/Deny a Token
  begin
     If(user is from a different domain)
     call Rolemap(credentials)
     find minimum rated role to access resource in the domain
```

```
        find the normalized global rating of that role GLR
        retrieve the Normalized global rating of user from certificate GLU
        if GLU$\geq$GLR
            return accept
        else
            return deny
        else
            retrieve user roles, rating and other credentials from database
        if(resource is from other domain)
            create token T containing the user details
            return T
        else
            find minimum rated role MR to access resource in the domain
        if(MR > user's role-rating) return accept
        else return deny
    end
```

```
Algorithm: Rolemap
Input : User Credentials / Token
Output : Token containing updated values
    begin
        if(user is from different domain or sub-Domain)
            Token T = call Rolemap(credentials)
            Add rating of the sub-Domain to
            the Global rating of the resource in the Token
            return T
        if(user is from same Domain)
            Token T = call Authorize(Credentials)
            Add Rating of Sub - Domain to T
            Return T
    end
```

4. Implementation Details

The Authorization Servers (AS1 and AS2) mentioned in Figure 2, work based on the architecture as described in Figure 4. It follows the algorithm shown as above. The Policy Enforcement Point (PEP) takes the credentials of the user and creates a request for authorization in the XACML format. This request is forwarded to the Policy Decision Point (PDP). The PDP checks the policies and sends back a reply in the same XACML format. PEP acts based on this reply and sends either deny or grant to the request. We have implemented this architecture for a single domain grid enterprise with indirect authorization (delegation) mechanisms. We have extended this implementation for cross-domain authorization by ranking the roles using role-mapping mechanism. We have also incorporated Role Based Delegation and Revocation models for multi-domain grid environments in the implementation mechanism.

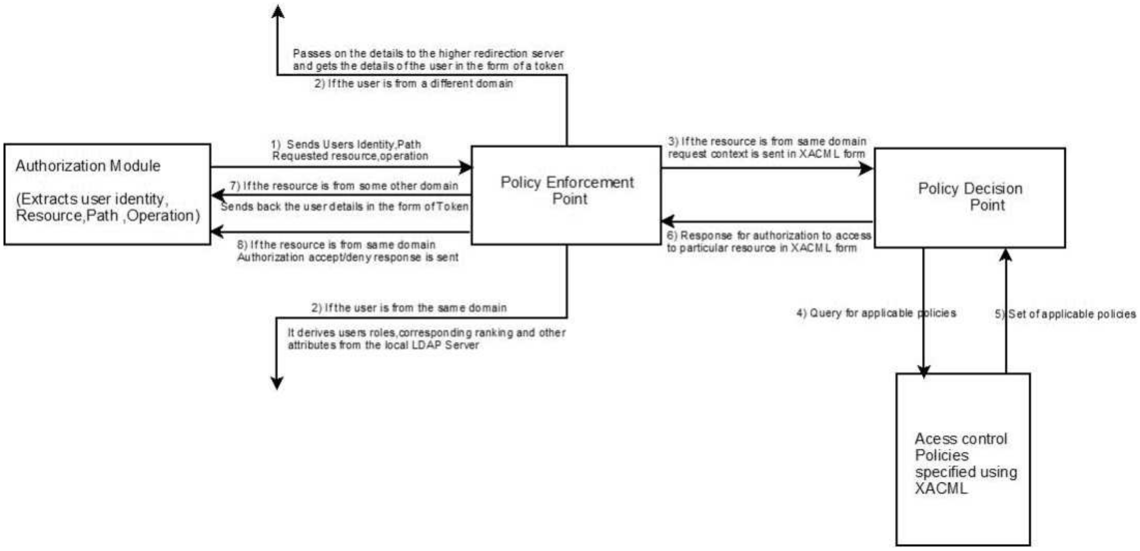
Passes on the details to the higher redirection server
and gets the details of the user in the form of a token
2) If the user is from a different domain
Authorization Module
(Extracts user identity, Resource,Path ,Operation)
1) Sends Users Identity,Path Requested resource,operation
7) If the resource is from some other domain
Sends back the user details in the form of Token
8) If the resource is from same domain Authorization accept/deny response is sent
Policy Enforcement Point
3) If the resource is from same domain request context is sent in XACML form
6) Response for authorization to access to particular resource in XACML form
Policy Decision Point
2) If the user is from the same domain
It derives users roles,corresponding ranking and other attributes from the local LDAP Server
4) Query for applicable policies
5) Set of applicable policies
Acess control Policies specified using XACML

Fig. 4. Authorization Server Architecture

5. Conclusion and Future Work

Our architectural framework proposes a methodology to solve the issue of security in grids through cross domain grid access control with delegation and revocation mechanisms with additional scalability. Also the proposed role mapping architecture makes it possible for grid nodes across domains to interact and authorize users for resource access. This architecture supports reusability of role-ranking mechanism, as the token once created between two nodes can be used for future interactions between them. Domains can also formulate additional access control policies like giving access to only semantically sensible organizations for that particular resource, apart from just comparing the global ranking of the user. We plan to continue our research for more fine grained access control features and incorporate better normalization techniques for role-ranking.

References

[1] Ajith Kamath, Ramiro et.al, "User-Credential Based Role Mapping in Multi-domain Environment", Proceedings of the Privacy, Security, Trust (PST), 2006.

[2] Basit Shafiq, James B.D. Joshi et.al, “Secure Interoperation in a Multidomain Environment Employing RBAC Policies”, IEEE Transactions on Knowledge and Data Engineering, Vol 17, No.11, November 2005.

[3] eXtensible Access Control Markup Language, Version 2.0, OASIS Standard, February 2005, http://docs.oasisopen.org/xacml/2.0/access control-xacml-2.0-core-spec-os.pdf

[4] Ezedin Barka, Ravi S. Sandhu, “Role-Based Delegation Model/ Hierarchical Roles”, In 20th Annual Computer Security Applications Conferences,Tucson, AZ, USA, pp 396-404, December 2004.

[5] G. Geethakumari, Atul Negi, V. N. Sastry, “Indirect Authorization Topologies for Grid Access Control”, In 9th International Conference on Information Technology, Bhubaneswar, Orissa, India, pp 186-187, December 2006.

[6] I. Foster, C. Kesselman, “The Grid: Blueprint for a new Computing Infrastructure”, Morgan Kaufmann Publishers, ISBN 1-55860-475-8, 1999.

[7] Ian Foster, “What is the Grid? A Three Point Checklist”, Grid Today, July 2002, http://www-fp.mcs.anl.gov/ foster/Articles/WhatisTheGrid.pdf.

[8] Ian Foster, Carl Kesselman, Steven Tuecke, “The Anatomy of the Grid:Enabling Scalable Virtual Organizations”, In 1st IEEE International Symposium on Cluster Computing and the Grid, Brisbane, Australia, pp 6-7, May 2001.

[9] Jiageng Li, David Cordes, “A Scalable Authorization Approach for the Globus grid system”, Future Generation Computer Systems Archive Vol 21, Issue 2, pp 291-301,February 2005.

[10] Liang Chen and Jason Crampton, “Inter-domain Role Mapping and Least Privilege”, Proceedings of the Symposium on Access Control Models and Technologies (SACMAT), 2007

[11] Laura Pearlman, Von Welch, Ian Foster, Carl kesselman, Steven Tuecke,“A Community Authorization Service for Group Collaboration”, IEEE 3rd. International Workshop on Policies for. Distributed Systems and Networks, 2002.

[12] Longhua Zhang, Gail-Joon Ahn, Bei-tseng Chu, “A Rule-Based Framework for Role-Based Delegation and Revocation”, In ACM Transactions on Information and System Security, pp 404-441, August 2003.

[13] Marty Humphrey, Mary R Thomson and Keith R Jackson, “Security for Grids”, Proceedings of the IEEE, Vol 93, No 3, pp 644-652, March 2005.

[14] Messaoud Benantar, “Access Control Systems”, Springer Publications, 2006.

[15] Miguel L. Bote-Lorenzo, Yannis A. Dimitriadis, Eduardo Gmez-Snchez,“Grid Characteristics and Uses: A Grid Definition”, In European Across Grids Conference, Santiago de Compostela, Spain, pp 291-298, February 2003.

[16] R. Alfieri, R .Cecchini et al, “From Gridmap-File to VOMS: Managing Authorization in a Grid Environment”, Future Generation Computer Systems, Vol 21, pp 549-558, 2005.

[17] Ravi S. Sandhu, David F. Ferraiolo, D. Richard Kuhn, “The NIST Model for Role-Based Access Control: Towards a Unified Standard”, ACM Workshop on Role-Based Access Control, 2000, pp 47-63

[18] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, Charles E. Youman,“Role-Based Access Control Models”, In Proceedings of International SemanticWeb Conference, Sanibel Island, Florida, USA, pp 706-721,20-23 October 2003.

[19] Von Welch, Frank Siebenlist, Ian T. Foster, John Bresnahan, Karl Cza-jkowski,Jarek Gawor, Carl Kesselman, Sam Meder, Laura Pearlman,Steven Tuecke, “Security for Grid Services”, In 12th International Symposiumon High-Performance Distributed Computing, Seattle, WA, USA, pp 359-368, 22-24, June 2003.

[20] Weizhong Qiang, Hai Jin, Xuanhua Shi, Deqing Zou, Hao Zhang, “RBGACA:A RBAC

Based Grid Access Control Architecture", In 2nd Grid and Cooperative Computing, Shanghai, China, pp 487-494, December 2003.