An Access Control Framework for Grid Environment

Seema*
Dept of CSE, Amity Unversity,
Gurgaon, Haryana, India
mudgil.seema@gmail.com

Sarbjeet Singh
CSE, UIET, Panjab University
Chandigarh, India
sarbjeet@pu.ac.in

Dolly Sharma
Deptt. of IT, BUEST
Baddi, HP, India
dollysharma83@ymail.com

Abstract-
Grid infrastructures provide an environment for high performance computing by combining heterogeneous, widely distributed computational resources into a single cohesive computing infrastructure. One of the key goals of grid computing infrastructure is to provide an environment where users can share resources across heterogeneous domains. In such an infrastructure security is the major concern. Security itself is composed of various sub-issues like authentication, authorization, single sign-on, data integrity, confidentiality etc. Authorization deals with the issue like who is authorized to access which resource and under what condition i.e. controlling access to resources based on conditions. An authorization system must address different concerns like scalability, fine-grain access control, context based authorization, continuous decision enforcement. This paper proposes an authorization framework which addresses above issues in making access decision on the resource. This paper also discusses a prototype implementation of the framework and results show that framework is workable and addresses all issues related to authorization and access control.

Keywords: Grid computing, security, authorization, access control and Access Control Framework.

I. INTRODUCTION AND BACKGROUND

Grids provide efficient and scalable access to distributed computing capabilities and enable resource sharing among heterogeneous domains over wide area networks. The problem that led to the development of grid technologies occurred first in scientific communities which needed access to computing and storage resources exceeding the locally available resources. Grid computing refers to the automated sharing and coordination of the collective processing power of many resources. A community of users and resource providers sharing their resources in such a way is called a Virtual Organization (VO), [Foster (2001)]. VO provides ability to share and aggregate computational capabilities which are spread over heterogeneous wide area networks and deliver them as service. Such system presents security problems i.e. authentication, authorization, data protection, delegation of access rights, single sign-on, and privacy etc,[Foster *et al.* (1998)],[Humphery *et al.* (2005)]. Security requirements in grid environment have been stated in [Nagaratnam (2003)].The basic security components are comprised of mechanisms for authentication, authorization, and confidentiality of communication between grid computers. Without this functionality, the integrity and confidentiality of the data processed within the grid would be at risk.

The Grid Security Infrastructure (GSI) [Foster and Kesselman. (1999)] has been accepted as the primary authentication mechanism for the Grid. It has been developed as part of the Globus project . The GSI leverages a Public Key Infrastructure (PKI) for mutual authentication among users, services and resources based on X.509 public key certificates [Housley *et al.* (2002)]. The GSI released with the Globus Toolkit 3 (GSI3) [Welch V. *et.al.* (2003)] is based on and largely compatible with the earlier GSI implementations but has been improved and augmented with new features aimed at securing a service based architecture. The new features support Web Services Security specifications, such as SOAP with XMLSignature. Though GSI addresses important issues related to security but authorization is not fully addressed.

Authorization deals with the verification of an action that an entity can perform after authentication is performed successfully. In a grid, resource owners will require the ability to grant or deny access based on identity, membership of groups or virtual organizations, and other dynamic considerations. Thus policies must be established that determine the capabilities of allowed actions. Thus the issue of "who has got the access" is authorization and "under what condition" is access control [Chakrabarti (2007)]. Thus term authorization refers to a statement or combination of statements that prove a subject's right to access a resource, or to describe the act of validating that a subject has such a right. Computational grids and other heterogeneous, large-scale distributed systems require powerful authorization mechanisms. Authorization system should be scalable i.e. increased number of users/resources/domains should not create an overhead. Access control policy should be flexible and must provide fine-grain access control on resources. authorization mechanism should consider dynamic nature of environment and authorization decision should be a continuous process i.e. changes in the system must be reflected e.g. consider a scenario where a user sends an access request for file that contains confidential data and access is granted as user is accessing from a secure network connection . If during the access operation user changes location and now is in an insecure location then this change should be reflected in the decision process. Various mechanisms have been proposed by researchers to solve authorization and access control in grid environment [Foster *et al.(*1998),[Thompson et al.(1999)],[Thompson *et al*.(2001),[Pearlaman *et al.* (2002)],[Chadwick and Okento(2002)],[Alfieri *et al*. (2003)],[Lorch *et al.(*2003)]and [Zhang *et al*.(2006)] . Earliest mechanism implemented in GSI is based on ACL (access control list). It uses grid-map file on each resource. An access request is authorized if a resource has an entry in its grid map file (its ACL) for the (authenticated) global identity presented with the request. The global user-id is then mapped to a corresponding local user-id and finer grain authorization is left to the local resource operating system mechanisms. Through GSI mechanisms only coarse grain access decisions are possible and this approach is not scalable as it is not feasible to have users' accounts on each resource also it does not consider dynamic nature of grid environment.

Classification of these authorization mechanisms has been drawn on the basis of factors scalability, fine-grain authorization, context-based authorization and decision continuity [Seema *et al.* (2009)]. In this paper we propose a framework for access control which provides fine-grain access control, context-based authorization and decision continuity. The proposed framework is scalable as well.

II. FRAMEWORK ELEMENTS

An access control framework can be defined as a system that controls access to services/resources made by grid users based on authentication, authorization attributes of subjects, attributes of objects/resource as well as system attributes which conforms to policies. Each entity i.e. subject and object/resource is identified by its attribute. Subject's attribute are divided in two categories mutable and immutable. Mutable attributes are those which may change during access operation e.g. location, usage status etc. Immutable attributes are those which are independent of access operation e.g. identity of user. The novel feature of the framework is that each domain has an attribute authority which acts a filter which filters the rights of the requesting user for the target site. Since there is no central authority which provides attributes to user and also there is no need to have each user to have account on the target resource.

In the framework following elements are used.

***2.1 Virtual Organization*:** VO is formed by participation of different physical organizations called domains (DO). This participations forms co-operation among participating domains where users from one domain can access services/resources from other domains and service provider expose their services to other domains

***2.2 Subject (SU)*:** Subject is an entity that wants to access services/resources. It can be a user, service or any other entity on behalf of user/service.

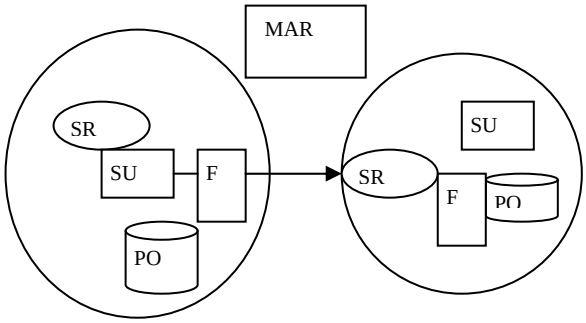

Figure 1: Elements of Framework

2.3 Service/Resource (SR)*:* Service/resource is the object which is accessed by user e.g. CPU, Storage device, data, instruments etc.

2.4 Service Policy (SP)*:* It is a set of rules associated with service/resource. Subject must conform to service policy in order to access that service.

2.5 Policy database (PO): It is policy store maintained by each domain that contains access rules i.e. SP (service policies) associated with services in that domain. While evaluating access request for a service policy associated with that service is checked for adherence of subject to service.

2.6 Filter/Privilege Authority*:* The rights and privilege are different in different domains. This component filters rights of the subject for the domain to which it is making request. When a virtual organization is formed then various participating organizations register themselves with each other so that they can interact and access services of other domains. During this process every domain informs other domain about rights that this domain gives to other domain. So every domain maintains a database about rights which other domains have given to it which acts as filter.

2.7 Mutable Attribute Repository: This is central repository which contains subject's mutable attribute e.g. usage quota (computation time, storage space) etc and system attributes e.g. time, location etc. These attributes are pulled by the authorization system while evaluating policy. These attributes can be updated during or after the access is granted which leads to decision continuity property of the framework.

Fig. 1 shows an environment consisting of two domains (DO1 and DO2) with other elements of framework. In the diagram square represents subjects, circle represents service/resource, rectangle with text FA represents filter authority, rectangle with text MAR represents mutable attribute repository. When subject request to access a service SR from other domain then first it gets authorization assertion from FA i.e. its rights are filtered for that particular domain to which it is making request. If subject is authorized for the target domain then request is forwarded; at the target domain access control framework checks for the conformance of subject to the policies of the resource/service. The flow of information from requesting domain to target domain as well as detailed description of the framework is explained in section III.

III Access Control Framework

In the access control framework each entity is presented as a set of attributes for e.g. the subject attribute can be group membership, role assignment and the object (resource) attribute can be its usage status. All these subjects and resources are part of some domain which in turn is part of VO. In order to access some service subject first has to gather credential i.e. authentication and authorization credential.

3.1 Obtaining Credentials

In order to access service/resource user needs to find authentication requirement of service. Since there can be different authentication requirements of services e.g. X.509, Kerberos tickets etc. So each domain maintains different credentials of subjects in the database. Requesting subject first finds out authentication requirement of resource/service (X.509, Kerberos ticket etc.) and then access the required credential from the database. In our framework authentication requirement is assumed to be X.509. There is one root authority on which all the participating domains agree. This root authority delegates its right of issuing certificate to individual domain. Root certificate authority issues certificate to domains and domains issue certificate to users.

After obtaining authentication credential/identity certificate requesting subject contacts Filter/Privilege authority in its home domain for obtaining authorization assertion which states whether user has access to the target domain. Thus Filter/privilege authority filters rights of user for the target domain. This filter/privilege authority maintains a database where information about privileges/rights is stored which other participating domains have allowed to requesting domain as in figure 3.1. Since this information is distributed in each participating domain and it is not centralized so the proposed framework is scalable. After obtaining the entire credentials subject makes access request to the target domain.

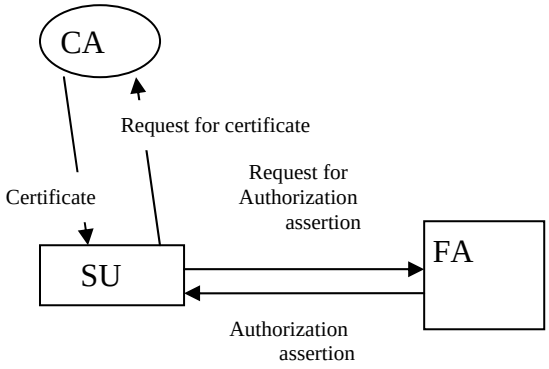

Fig. 3.1 collecting credentials

3.2 Reaching Authorization decision

The authorization decisions are made on the basis of attribute values. The attributes are divided in two categories mutable and immutable attributes. Immutable attributes are those attributes which do not change due to access which are pushed with the access request where as mutable attributes change due to the side-effect of access for e.g. usage quota allocated to a subject. These mutable attributes are stored in mutable attribute repository as shown in the figure 3.2. Subject that request for a service/resource sends request to the target site with certificate from certificate authority which authenticates subject's identity at target site. Request also includes persistent attribute of the subject i.e. role or group membership. At the target site request is intercepted by PEP (policy enforcement point) .PEP validates certificate which comes with request. If certificate is valid then PEP forwards request to PDP (policy decision point) for making access decision. After getting request from PEP, PDP retrieves corresponding policy from the policy store which matches with the subject's, resource's and action's values. If policy requires some mutable attributes of user then it fetches from mutable AR and object's attribute from RM. Then PDP evaluates policy and decides upon request and responds to PEP which then enforce decision

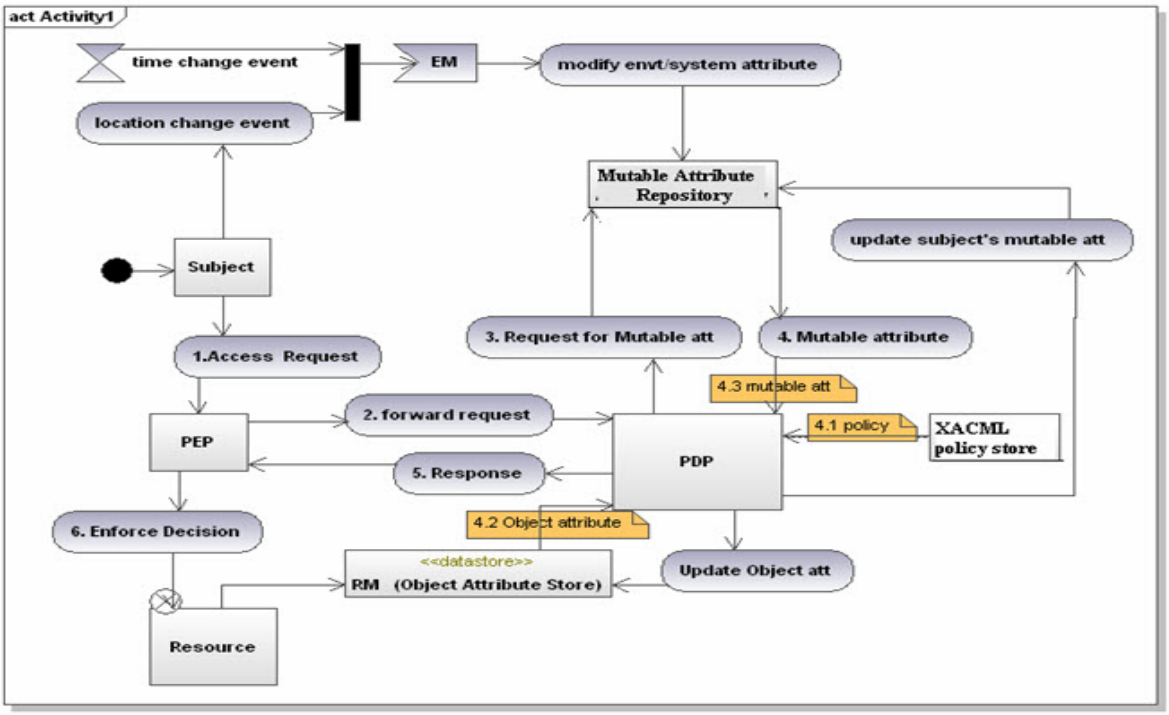

Figure 3.2 Reaching Authorization Decision

The access control framework supports attribute mutability and continuous enforcement of the access decision based on the subject and object attributes. Attribute mutability denotes changes in attributes of subject and resource that happens when subject requests for accessing resource. Updates on the subject and object attributes are performed by

PDP where as environment attributes updates do not depend on the usage session so they are performed by EM. Update actions performed on attributes are as follows.

Pre-updates*:* Pre-updates are performed when PDP approves access request based on the policy for e.g. a subject sends an access request for modification to a resource (write operation) which can be read by many but modification can be performed by only one subject at one time. If the subject adheres to policy for the operation to be performed then access is granted and the object attribute 'Status' is updated to 'write' which is designated by value "1" in our implementation.

Ongoing-updates: Ongoing updates are performed by some events in the system e.g. if during the usage process subject changes location. This change is tracked by EM component and is reported to PDP.
PDP re-evaluates policies and decision whether to continue or revoke the ongoing usage is made and enforced Decision continuity is based on the attributes updates i.e. whenever attributes are updated policy is re-evaluated and decision is made.

There are two components in the access control model namely RM (resource monitor) and EM (environment monitor) as shown in fig 3.2. These are important components of model and are explained as follows.

***EM (Environment Monitor*)**: This component captures any changes in the environment (e.g. location changes of user, time- change event etc). These changes are reflected in mutable attribute repository.

RM (Resource Monitor): This component stores resource attributes (e.g. status of the resource). When decision has to be made these attributes values are provided to PDP (policy decision point) which are used in evaluating policies associated with the resource

IV IMPLEMENTATION- A Case Study

The proposed access control framework is implemented in .NET environment using web services. For implementation a case study is taken in which three domains namely UBS, UIET and History are taken. These three physical domains co-operate to form a VO (virtual organization) called Core Grid as shown in fig 4.1. Each domain has subjects and services/resources. These services are exposed to other domains through discovery service which runs in CoreGrid.

.

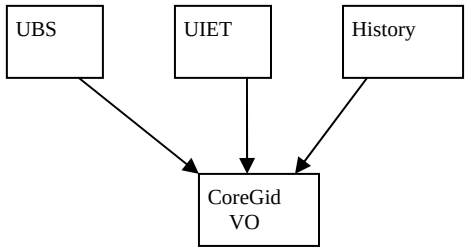

Fig 4.1 Physical domains and VO

Table 4.1 List of services

Domain	**Service**
UIET	Computation Service
UBS	Storage Service
HSTORY	Data Service

When a subject needs to access a resource it first contacts discovery service to find out in which domain it exists. This service can also provide details of authentication requirements of services if services implement different authentication methods but since in our framework we have only one authentication requirement i.e. X.509 so we have omitted this detail in current implementation. Thus in order to access any service, subject first enters into its home domain i.e. the domain of which he is a registered member. Each domain maintains database of its users, when subject enters its domain by providing its credential (username/password); its credentials are verified and subject is issued X.509 certificate that will authenticate him at target side. Then he searches for the services in VO by accessing discovery service in CoreGrid. After selecting a service for a particular action he makes an access request. Before

request is forwarded to other domain FA filters its rights for that domain. Each domain has a database of privileges that other domain has assigned to it. With this concept domains need not to keep track of access rights of subjects of other domains. When access request is made for a particular action then right is filtered at home domain only. If subject is authorized for that action then only its request is forwarded. Procedure for sending request is depicted in fig 4.2. At the target side request is intercepted by PEP (policy enforcement point) which verifies subject's authentication certificate. If subject is authenticated then request is forwarded to PDP. PDP retrieves policies from database by matching attributes of resource, action and subject

```
Procedure PEP
{
Call ValidateCertificate(Path)
//checks the certificate validity by accessing
      certificate
If (certificate==valid)
Call PDP (attributes of subject, resource, action)
}
```

Fig 4.2 Procedure for sending request

If subject complies with the policies then request is granted. Since there can be multiple policies/rules for service to be accessed therefore a procedure for combining results from different policies is needed. We are implementing deny-override rule combining algorithm of XACML [Godik *et al*. (2003).] model which means thet if one policy evaluates to "deny" then overall result is "deny". Procedures for evaluation performed by PDP, PEP are shown in fig 4.3 and 4.4 respectively. Procedure for rule combining is shown in fig 4.5.

```
Procedure PDP
{
Call getpolicies ("path")
// find  all policies applicable to the request
      based on attributes of subject, resource
      and store result in array r [ ].
Call combinepolicy (r [ ])
// combine result of all policies.
 return result;
}
```

Fig 4.3 Procedure for policy evaluation

```
Procedure validating user and sending request
{
Call IsValidUser ()

  If o.IsValidUser(Login1.UserName, Login1.Password)
Then   Response.Redirect("~/ToEnterDiscoveryServices.aspx")
Else  MsgBox("InCorrect UserName/Password")
         Response.Redirect("~/Login.aspx")
      End If
// if (user==valid) returns true; //if user is registered member of domain.

IsValidCertificateCredentials() // get identity certificate from domain authority.

Call SearchTheServices()

// redirects request to discovery service in coregrid which returns list of services in
      domains. User selects the service and make access request

Call   SendRequestToOtherDomain(attributes   of   subject,   resource,   action,
      authorization assertion)

//returns authorization credential from filter authority to enter into domain and calls
      PEP of other domain.
```

Fig 4.4 Procedure for sending request to PDP

```
Procedure combinepolicy(r [ ])
{
for (i=0; i<lengthof(r); i++)
{ If (r==deny)
return deny;
else continue;
}
```

Fig 4.5 Procedure for policy combining

If, during access operation, some environmental changes occur e.g. location change event of subject and this change is reflected in EM component of the model and any mutable attribute change is updated in MAR. These attributes are pushed to PDP and polices are re-evaluated to see whether user complies with the policies or not and decision is enforced.

V Results and Discussion

A Grid Environment is set up using ASP.NET with Web Services. Three domains are created which are three different projects namely UBS, UIET, HISTORY. A CoreGrid domain is one which acts VO having all three domains. The database is created in Microsoft SQL Server 2005. The resulting comparison charts have been drawn using Microsoft Excel package. Framework is scalable as there is no central authority. Each domain manages its own resources and access policies. PDP is also distributed in individual domain. There is only root authority which issues certificates to domain and users are issued certificate in their home domain only. When access request is made to a resource in other domain root certificate with domain certificate is sent. So user need not to ask central root authority i.e. if number of users requesting for services increases then it does not put bottleneck at central authority. The proposed framework is tested against various policies. Performance of the system is measured by varying number of policies. System performance is measured through interaction of user in one domain who tries to access service in other domain under condition which shows attribute mutability, decision continuity property through various cases shown in table 5.1.

Table 5.1 Interaction of user in domain with services in other domain

Case No	**Description**
Case 1	UBS domain user accessing History domain service
Case 2	UIET domain user accessing History domain service
Case 3	UBS domain user accessing UIET domain service

5.1 Case 1: UBS domain user accessing History domain service

A UBS domain user/subject having role as student logs in and sends request to read DataService of History domain. Before its request is sent to History domain user gets authorization assertion from Filter Authority of UBS. To provide authorization assertion FA of UBS uses permission table

Table 5.2 Permission table for UBS

Domain	**Permission**
History	Write
UBS	Execute

As subject request for reading therefore it gets authorization assertion as “permit” and request is sent to PEP of History domain. PEP validates certificate and sends request to PDP. PDP checks policies from policy store and result

is sent back to PEP. If a user request for write action on data service then its authorization assertion is “Deny” and request is not sent to PEP. Assertion based on requested action is tabulated in table 5.3.

Table 5.3 Assertion from FA

Action	Assertion
Read	Permit
Write	Deny

Performance of system is measured by noting time taken in processing request by PEP and PDP.

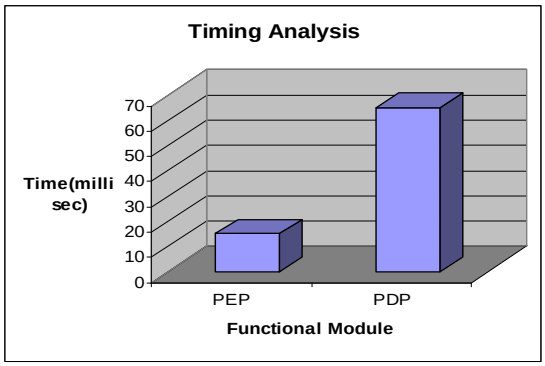

Figure 5.1: Timing Graph UBS domain user accessing History domain service

5.2 Case 2: UIET domain user accessing History domain service

This case shows how attribute mutability is implemented. RM of the History domain stores the attribute of DataService i.e. “in-use”. Initially its value is ‘0’ which means that it is not in use. If request for “write” action is permitted then its value is updated to ‘1’.

In this case subject with role as teacher logs in and sends request to access (write) into DataService of history domain. When user logs in it gets validated and certificate is obtained from the domain then its rights are filtered from the FA which is implemented through permission table of FA, table 5.4. Since user is requesting for write operation it gets authorization assertion as “grant” and request is sent to other domain. There PEP intercepts request and validates certificate and request is sent to PDP. PDP retrieves policies from the policy store and policies are evaluated. Since subject complies with the rules to access resource permission is granted.

Table 5.4 Permission table for UIET

Domain	Permission
History	Read
UIET	Execute

Since request is for “write” operation then mutable attribute of resource, which is DataService, is updated after access is granted. Thus status of resource is set to 1 as shown in figure 5.2.

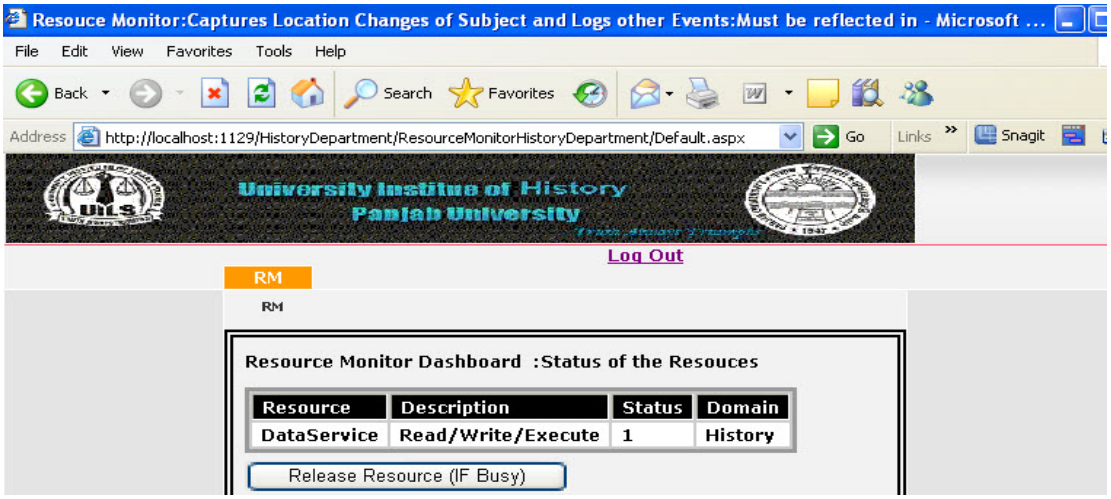

Figure 5.2 Updated resource monitor

5.3 Case 3: UBS domain user accessing UIET domain service

This case shows context-based authorization and decision continuity property of the model.

In this case a UBS domain user sends request to access Computation Service of UIET domain for action “execute”. It gets authorization assertion from FA of its home domain UBS i.e.” grant” and request is sent to PEP. PEP validates incoming request and sends request to PDP. PDP checks policies and since subject confirms to policies request is granted. Permission table for UBS domain is shown in table 5.2. Some of the policies attached with service are

- UBS domain user is permitted.
- If subject changes location then request is aborted.
- Service can be accessed from Mon-Wed.
- Service can be accessed during 9am -5 pm

Time taken in processing the request is noted and is shown in figure 5.3.

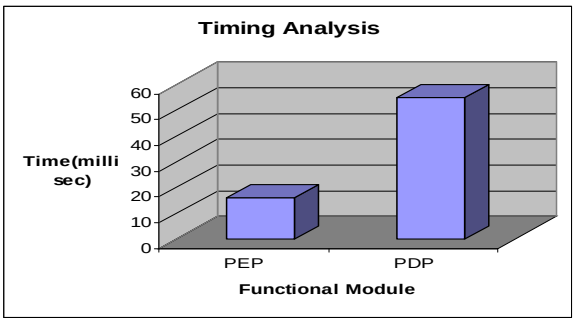

Fig 5.3 Timing graph for UBS’s subject/user accessing UIET domain’s service.

If subject changes location from UBS domain to History domain as shown in figure 5.4 then policies are re-evaluated as EM captures this location change event in the environment which in-turn changes mutable attribute of subject in MAR and these changed attributes are pushed to PDP. This change in location does not adhere to policy for resource access i.e. UBS user is permitted gets failed as subject changes location to History which is considered as in-secured by Computation service of UIET. Thus access is aborted. Since policies are re-evaluated thus processing time is taken only by PDP and not PEP.

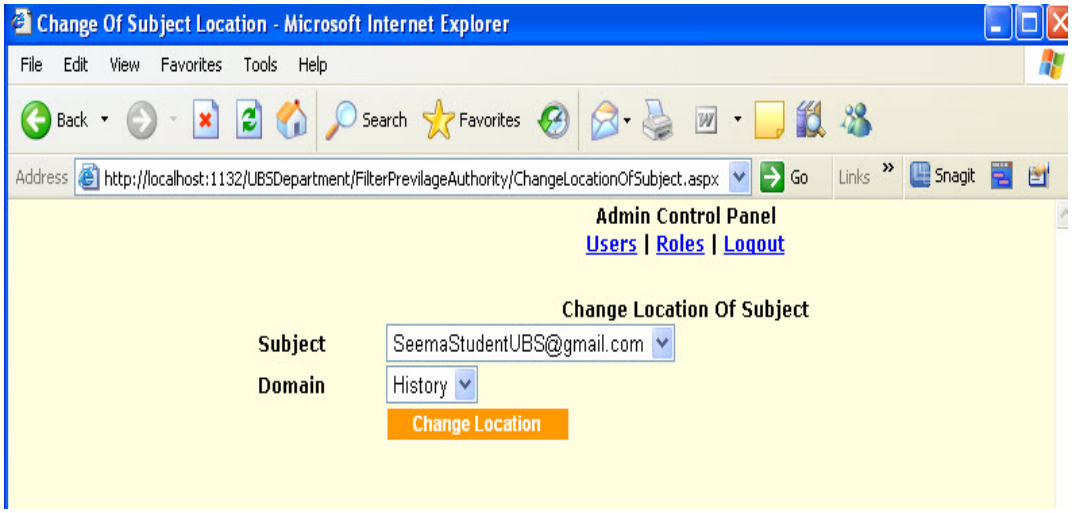

Fig 5.4 Location change event of subject in UBS domain

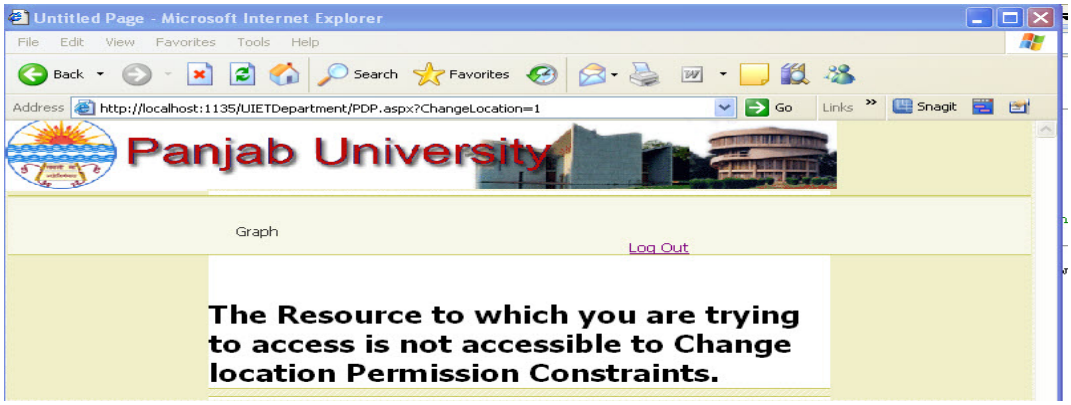

Fig 5.5 Access denied due to change location

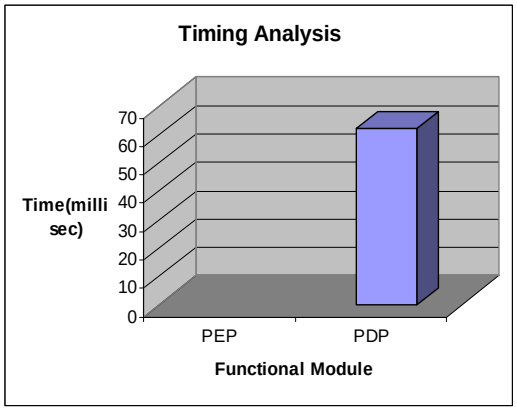

Fig 5.6 Timing graph for location change

5.4 Performance of system with increasing number of policies

Framework is tested on one more scale where policies are increased gradually and time taken in evaluating these policies is noted down. As the number of polices are increased, time taken in evaluating these policies increases. But this increase is not exponential. Time increases linearly with increasing number of polices. Thus framework is efficient and does not put performance overhead as numbers of policies are increased to access resource/service. Timing graph for the above scenario is shown in figure 5.7.

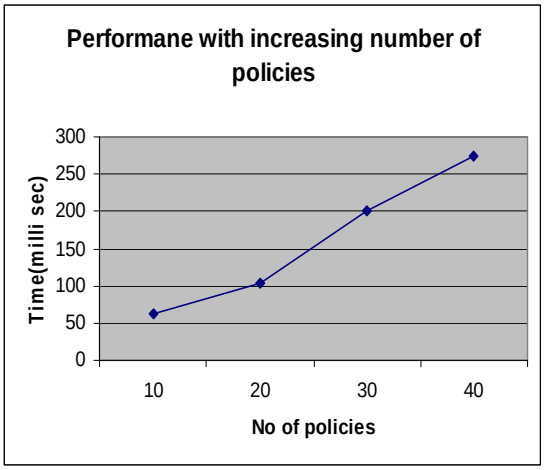

Fig 5.7 Performance with increasing number of policies

VI Conclusion and Future Scope

Proposed authorization framework is tested for various scenarios and results shows that it satisfies parameters i.e. scalability, decision-continuity and context-based authorization. Framework is scalable as there is no central point of interaction. Each domain manages its resources and has control over them. PDP is also distributed in the respective domains thus there is no single point of bottleneck. It provides fine-grain authorization and policy writing as access control policies is written in XACML which is XML standard. It captures environmental changes and supports decision continuity based on changes if during access any environmental changes occur e.g. location change as in our implementation then policies are re-evaluated and new decision is enforced.By increasing number of policies, proposed system shows approximately linear growth in evaluating policies which shows proposed system is efficient and scalable.

The framework is about controlling access to resources based on policies; this can be extended to include accounting module which will calculate charges for accessing services. One aspect of grid usage is not included in current implementation i.e. delegation where the service/resource acts on the behalf of subject in order to process request. This is the situation where service requires accessing another service/resource that may be lying in other domain. Framework can be extended to include delegation and delegation policies. A GUI toolkit can be developed for policy writing so that a novel user(resource provider) can also use it for expressing access control policies.

VII. References

[1] Alfieri R, Cecchini R.,CiaschiniV,Dell L.,Á.Frohner, A. Gianoli, K. L˝orentey and F. Spataro, (2003) :"VOMS, an Authorization System for Virtual Organizations", in Proceedings of the 1stEuropean Across Grids Conference, Santiago de Compostela.

[2] Alfieri R, Cecchinib R, Ciaschinic V, L dell'Agnellod, A. Frohnere, K. Lorenteyf, and F. Spatarog. , (2005): "From gridmap-file to voms: Managing authorization in a grid environment". *Future Generation Computer Systems 21*

[3] Chadwick D., Okento (2002):"The PERMIS X.509 role-based privilege management infrastructure", Pre-print version of Future Generation Computer Systems. 936 (2002)

[4] Chakrabarti Anirban (2007):"Grid Computing Security", XIV, 332 p. 87 ISBN: 978-3-540-44492-3

[5] Dey, A.K. Abowd, G.D. (2000) Towards a Better Understanding of Context and Context-Awareness.CHI

[6] Dey, A.K (2001)." Understanding and Using Context". Personal Ubi Comp 5 1, 4–7

[7] Ferraiolo D. F. , J. F. Barkley and D. R. Kuh (1999), "A Role Based Access Control Model and Reference Implementation Within a Corporate Intranet", ACM Transactions on Information urity,2(1): pp.34-64,

[8] Foster I. C. Kesselman; G. Tsudik, S. Tuecke (1998) : A Security Architecture for Computational Grids. Fifth ACM Conference on

[9] Computers and Communications Security,

[10] Foster I., C. Kesselman, S. Tuecke, (2001):"The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International J.Supercomputer Applications*, 15(3)

[11] Foster I., and C. Kesselman, (1999)"Globus: A Toolkit-Based Grid Architecture" in "The Grid, Blueprint for a Future Computing Infrastructure", I.Foster, and C. Kesselman, Editors, Morgan Kaufmann, San Francisco. pp. 259-278

[12] Godik S., Moses T, et al, (2003) :eXtensible Access Control Markup Language (XACML) Version 1.0", OASIS Standard

[13] Housley R, W. Polk, W. Ford, D. Solo (2002)"Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", Internet RFC3280, Internet Engineering Task Force, Public Key Infrastructure Working Group,

[14] Humphery Marty . Mary R. Thompson, Keith R.Jackson (2005)," Security for grids", Invited paper , Proceedings of IEEE , 93(3):644-652.

[15] Lorch M , D.B Adams, D. Kafura , M.S.R. Koneni, A. Rathi, and S. Shah.(2003.)" The PRIMA system for Privilege Management, Authorization and Enforcement in Grid Environment." In the proceedings of 4th IEEE International Workshop on Grid Computing,

[16] Nagaratnam N ,.Janson P, Foster I (2003): "Security Architecture for Open Grid Services",GGF OGSA Security workgroup

[17] Park J , Sandhu R (2004) :The $UCON_{abc}$ usage control model . ACM Transaction on Information and System Security, 7(1)

[18] Pearlman L. et al (2002):"A Community Authorization Service for Group Collaboration", IEEE Workshop on Policies for Distributed Systems and Networks

[19] Sandhu R, Coyne E., Feinstein H., Youman C., (1996):Role-Based Access Control Models, IEEE Computer, 29(2): pp.38-47

[20] Seema Dolly Sharma, Sarbjeet Singh, Amandeep Verma, ,(2009): Classification of Authorization Systems in Grid Environment" 2009 IEEE International Advance Computing Conference (IACC 2009) ,Thapar University, Patiala. Pp 3093-3098.
[21] THOMPSON M, et.al., (1999): Certificate-based Access Control for Widely Distributed Resources, *Proceedings of the Eighth Usenix Security Symposium*
[22] Thompson M, Essiari,Mudumbai A.,(2001): "Certificate-based Authorization Policy in a PKI Environment," ACM Transactions on Information and System Security (TISSEC), ACM 1073-0516/01/0300-0034
[23] Vollbrecht J et al (2000):AAA Framework, Internet RFC2904, Internet Engineering Task Force, Network Working Group
[24] Welch V, Foster I, Kesselman C, Mulmo O,. Pearlman L, Tuecke S, Gawor J,(2004): High Performance Distributed Computing Annual PKI R&D Workshop
[25] Welch V. et.al., (2003): "Security for Grid Services**",** 12th Int. Symposium on High Performance Distributed Computing
[26] Zhang X,.Parashar M.,(2003):Dynamic Context-aware Access Control for Grid Applications∗ In the proceedings of 4th International Workshop on Grid computing
[27] Zhang X., Nakne M., Michael J., Sandhu R.(2006): A Usage-based Authorization Framework for Collaborative Computing System. Proceedings of eleventh ACM symposium on Access Control Model and technologies