CHAPTER 1

INTRODUCTION

1.1 Background

Distributed Environments are touching new heights, becoming more useful, popular and more complex with the emergence of service oriented architecture and computing technologies. These technologies aim to enable large scale resource sharing. Security is a big and challenging issue in these environments as it involves the federation of multiple heterogeneous, geographically distributed autonomous administrative domains. The dynamic and multi- institutional nature of service oriented environments like grid and web introduces several challenging security issues that require new technical approaches. The framework is intended to provide a simple, powerful, flexible and scalable authorization infrastructure for services exposed in a large scale distributed environment.

Communication is an essential part of life. When we talk about the communication actually we talk about the digital communication as the internet has evolved as the new medium of communication for the digital information. This digital form of communication is different from the traditional medium of communication. In the digital form of communication the role of authentication and authorization is very important. Each and every resource is placed online and everyone has equal access to internet connection. So it may be possible that the users unknowingly or intentionally change, modify or delete the information. So every user who wants to access any resources in the network should be authenticated first depending on the requirements. Now based on the authentication the owner of the resource decide whether this user is authorized to use the requested resource or not. Every authenticated user is not necessarily authorized to use the resources. The authorization may also depend on the level of security.

With the wide use of Internet-based applications, security in distributed systems becomes a serious issue to individuals, companies and organizations. This makes computer

security a necessity to all computer users. Thus, many users use a security service that can be used to protect them from others. However, security services require extra computing resources because they are time-consuming. The main design principles of the system are that the system should provide the user with fast, efficient, stable and flexible security service. The agents interact cooperatively in a distributed environment and collectively act on behalf of the users to provide them with a secure environment. In addition, security is achieved at different levels. At the first level, the user is authenticated. At the second level, the user is authorized. At the third level, the service provides encryption, decryption, digital signature and verification.

The accurate authentication of users is one of the main important issues in distributed information systems. An efficient authentication can accurately verify the identity of a user and then allows the user to access some pre-defined resources, such as reading a file or executing an application. Usually an individual can be authenticated by his/her knowledge of a system or his/her physical characteristics. An assigned password is a traditional way for an individual to be authenticated by showing his/her knowledge of the system. Alternatively, keystroke biometrics is another way to perform authentication by human physical characteristics in terms of individual typing patterns.

In addition to providing basic security requirements like authentication, authorization, confidentiality and integrity, the security infrastructure for grid and web must also be able to support more advanced security features like dynamic delegation of access rights, single sign-on/sign-off, dynamic establishment of trust relationships among multiple domains, privacy and policy related security issues in federated environments etc. There are several factors that make security hard e.g. user population and resource pool is large and dynamic, resources have different authentication and authorization requirements, computations span over multiple domains, users have different roles/privileges in different domains etc All these factors make authorization a big and challenging issue. Authorization, in simple terms, deals with issues like who can access which resources/services under which conditions. There are many authorization mechanisms e.g. role based authorization, rule based authorization, and identity based authorization

etc. But these authorization mechanisms alone cannot satisfy the access requirements of distributed services as the access depends on many other factors like privacy requirements of the requester, authentication requirements of the service, trust relationship with the requester, authorization and management policies among participating parties etc. Authorization in a distributed environment should be determined as a result of evaluating the request of an authenticated user against various policies like privacy policy, trust policy, authorization policy etc. Many authorization mechanisms for large scale distributed systems like grid and web ignore one of the components from privacy, trust and policy. This work emphasize that these are vital components and must be integrated into the access control framework of security architecture to make it more effective and useful [1, 2].

In a distributed system, in order to keep message confidentiality and integrity, users are provided with different security measures such as encryption, decryption, digital signature and verification. It is desirable that user's messages are protected from being viewed by unauthorized people in transit. To achieve this goal, symmetric and/or asymmetric cipher is used to encrypt the message at the sending end and decrypt it at the receiving end. Digital signature and verification provide users with assurance that a message has not been tampered and came from a specific person. These security services are time-consuming and need massive computation. To solve this problem, a multi-agent based service mechanism is proposed. This proposed service mechanism includes a central controller that acts not only as the unique access point for external layers but also as the resource allocation center and several service providers that are able to cooperate using a pre-defined communication protocol to satisfy user's service requirement [1, 2].

1.2 Overview

The tremendous growth of the network-centered (Internet and Intranet) computing environments requires new architecture for security services. Computer crimes are growing rapidly in these environments. In addition, these computing environments are

open, and users may be connected or disconnected at any time. This makes computer security a necessity to all computer users.

During the last few years, it has become an increasing concern to the network administrators about how to control the users that are making use of computers networks. Several security technologies have recently emerged in order to provide access control mechanisms based on the authentication of users. Traditionally, network access systems have been based on login/password mechanisms to authenticate the different users requesting a network connection, which provides a very limited degree of security. Other systems follow a more advanced approach for mutual authentication based on X.509 identity certificates, therefore offering a stronger security solution which makes use of public key cryptography. These systems are especially useful for those Internet Service Providers (ISPs) which are concerned about the real identity of the requestor as a key element in order to make a decision. There are other organizations where the different members or users are classified according to their administrative tasks, the type of service obtained, or some other internal requirements. For example, in a university environment, users can be part of the administrative staff, professors, or researchers [1].

In industry, we can easily find hierarchical relationships comprising employees, managers, CEOs, etc. Even ISPs classify their customers according to the different type of contracts, which involves different types of connection properties and services. In these previous scenarios, the identity could not be sufficient to grant the access to the resource being controlled, since we should know the role being played by the user in order to offer the right service. Therefore, we need a system able to grant to the different users the set of attributes specifying those privileges or roles. This kind of systems is usually designed following the principles of the Role-Based Access Control (RBAC) model. In this way, when end users request a resource or service, the decision is taken according to the set of attributes assigned to them. For example, an ISP might state that only users showing a premium attribute will obtain a particular quality of service or network bandwidth [1].

It is worth noting that authentication is based on X.509 identity certificates and authorization is based on attributes, and therefore they constitute two completely independent operations. However, they are usually related since authentication represents the first stage of most access control systems, i.e., obtaining the user's identity. Once the identity has been securely established, it is necessary to infer whether the user is authorized to make use of the network. Different authorization proposals can be used in the above-mentioned application scenarios, as for instance SPKI, X.509 AC or SAML. Indeed, there are several projects that make use of these proposals, such as Liberty or Shibboleth, which have extended some well-known authentication solutions in order to provide authentication mechanisms in Web environments [1].

Generally the authorization is based on the SAML and the XACML standards, which will be used for expressing access control policies, authorization statements, and authorization protocols. The authorization proposal is mainly based on the definition of access control policies including the sets of users pertaining to different subject domains which will be able to be assigned to different roles in order to gain access to the network of a service provider, under specific circumstances. These policies are central elements in the system, and require the definition of some entities responsible for managing their lifecycle. Moreover, the starting point is a network scenario based on the Authentication, Authorization and Accounting (AAA) architecture where all the operations related to authentication, authorization and accounting were centralized [9].

1.3 Problem Definition

As more resources are being made available over the internet and intranet, Authorization is becoming a big and challenging issue. In a large scale service oriented computing environment where thousands of computers, storage systems, networks, scientific instruments and other devices distributed over heterogeneous wide area networks presents unique security problems that are not addressed by traditional client-server/distributed computing environments. The Main problems are:

- User population and resource pool is large and dynamic.
- Resources have different authentication and authorization requirements.
- Computations span over multiple domains.
- Users have different roles/privileges in different domains.

Nowadays many organizations share sensitive services through open network systems and this raises the need for an authorization framework that can interoperate even when the parties have no pre-existing relationships.

1.4 Objective

Having identified the main hindrances in an effort to establish a secure communications, the objective of this research is to solve the problem of resource accessibility by unauthorized persons by implementing the authorization framework.

The main objectives of this study are:

- To investigate about the need of authorization systems.
- To find out the requirements for an authorization system.
- To find out about the authorization policies and its mechanisms.
- To develop the concept to solve the problem of resource accessibility by unauthorized persons by implementing the authorization framework.

Generally, most systems used and seen in many organizations and companies have an authentication process which is normally done through user-id/password validation process which i am already familiar with but those systems are then controlled through some authorization process to which i am not familiar with. Thus, this leads me to perform an analysis to understand the need, use, techniques and implementation of authorization.

1.5 Thesis Organization

The rest of the material of the study is organized into subsequent six chapters. Chapter 2 does a theoretical analysis authorization along with its different elements and controls. Chapter 3 focuses on Shibboleth, the most commonly used authentication and authorization tool. Chapter 4 focuses on implementation of authorization showing how an authorization can be implemented in an organization. Finally, Chapter 5 concludes the result of the analysis.

CHAPTER 2

ANALYSIS OF AUTHORIZATION

2.1 Authentication Vs Authorization

It is easy to confuse the mechanism of authentication with that of authorization. It is important to draw the distinction between these two mechanisms. The main differences between these two mechanisms from one another can be describes as follows [1, 2]:
Authentication is the mechanism whereby systems may securely identify their users. Authentication systems provide answers to the questions:

- Who is the user?
- Is the user really who he/she represents himself to be?

An authentication system may be as simple as a plain-text password challenging system or as complicated as the Kerberos system. In all cases, however, authentication systems depend on some unique bit of information known (or available) only to the individual being authenticated and the authentication system - a shared secret. Such information may be a classical password, some physical property of the individual (fingerprint, etc.), or some derived data (as in the case of so-called smartcard systems). In order to verify the identity of a user, the authenticating system typically challenges the user to provide his unique information (his password, fingerprint, etc.) - if the authenticating system can verify that the shared secret was presented correctly, the user is considered authenticated.

Authorization, by contrast, is the mechanism by which a system determines what level of access a particular authenticated user should have to secure resources controlled by the system. For example, a database management system might be designed so as to provide certain specified individuals with the ability to retrieve information from a database but not the ability to change data stored in the database, while giving other individuals the ability to change data. Authorization systems provide answers to the questions:

- Is user X authorized to access resource R?
- Is user X authorized to perform operation P?

- Is user X authorized to perform operation P on resource R?

Thus, authorization systems depend on secure authentication systems to ensure that users are who they claim to be and thus prevent unauthorized users from gaining access to secured resources. This can be graphically depicted as shown in the figure below:

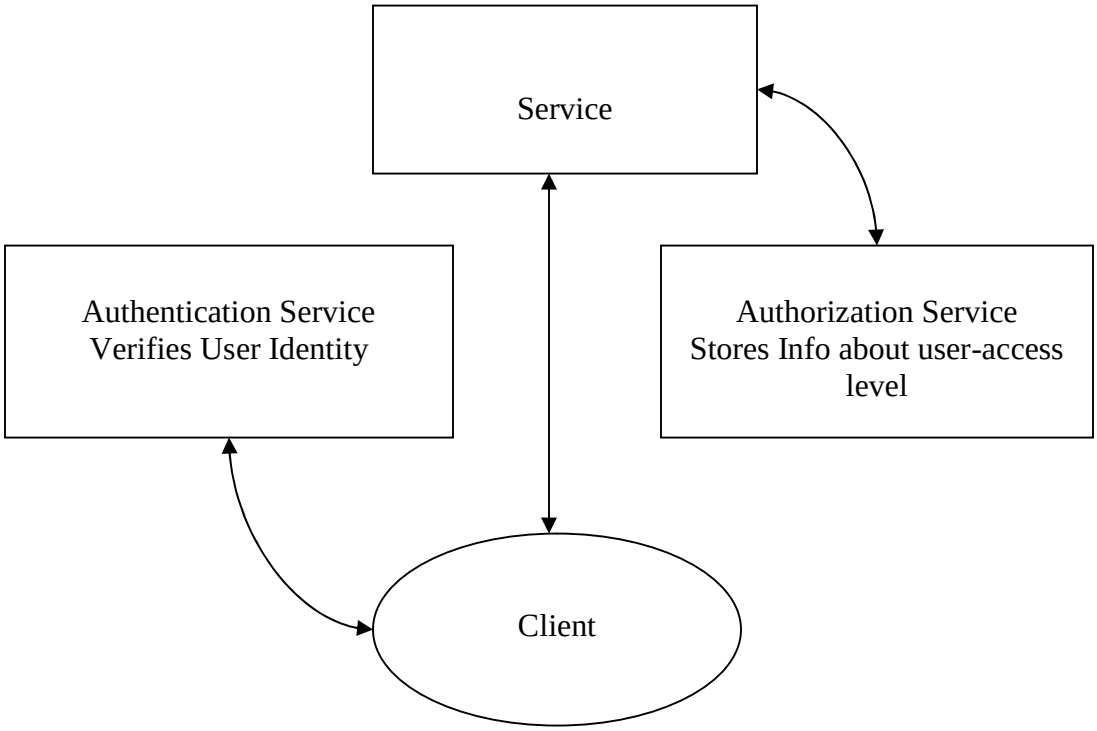

Figure 2.1:- Authentication vs. Authorization

Issues relating to the provision of authorization

The description of the authorization architectures shows that each tackles the problem in a slightly different way. This section outlines the important issues related to the construction of authorization architecture.

Privilege Distribution: Privilege distribution is the first important step in providing an authorization service. It is comprised of two separate tasks:

- Including a method for assigning privileges to users.

- Ensuring that these privileges are made available to the resource's AEFs.

To simplify the administration of user privileges a centralized approach is most commonly used. That is, user privileges are configured by one central authority. The advantage of this approach is that user privileges change frequently, and central administration allows the user privileges to be updated quickly. For example employee privileges typically change on a regular basis. This can occur when a new employee arrives (they are assigned privileges), when the employee is promoted or changes position (their privileges are modified) or when the employee leaves the organization (the privileges are revoked). Once a user has been assigned privileges, when the user accesses a protected resource, the AEF at the protected resource must have access to the user's privileges. This way the privileges can be compared with the access decision rules (normally ACLs) at the resource, to see if they satisfy the requirements.

The pull model is a term used to describe the process whereby an AEF communicates with an authority to get a user's privileges. This occurs whenever a user accesses a resource. The push model is a term used to describe the process whereby the user communicates their privileges to the resource's AEF when the access occurs [5].

There are several advantages of using the push model:

- There is no need for extra communications as the access decision can be taken immediately.
- There is only one access to the privilege attribute server by each client, compared to the many accesses that would be need by the target server.
- It is possible to access the server anonymously if the policy allows this. In the pull model, the server needs to know the client's identity to pull their privileges.
- It is possible to change the user's privileges according to the user's context (where did they log-in, which mechanism did they use to authenticate).
- The principle of least privilege: the client needs to provide to the server only the minimum of information that is needed to base an access control decision on (a system administrator can access a server without exposing their root privileges).

There is one disadvantage to using the push model:

- The target server is unsured that the privileges are valid at that point in time and have not been altered or revoked.

2.2 Access Control Mechanisms

In general, controlling the access to a resource (target) of an entity making an access request (initiator) is a multistage process, which requires the three following phases:

- Authentication of the initiator. Authentication means verifying the identity of a user, process or device; in the access control context, such an identity consists in a name (i.e. a finite string from a finite alphabet) that univocally distinguishes the entities that have been given permission to use (to some extent) a set of resources to which the target belongs.
- Authorization decision for the submitted request. Given the identity of the initiator, the access request and environmental or contextual information (e.g. current date and time), a granting or denying response must be returned according to the access control policy in force (and, where relevant, previously requested actions).
- Enforcement of the request. If the authorization decision phase grants the request, it must be enforced on the target.

With respect to access control mechanisms, the above results in two logically distinct functional modules: policy decision points (PDPs) and policy enforcement points (PEPs). PDPs are the modules which perform authorization decisions; whereas PEPs carry out the access decisions made by the PDPs, performing both the authentication and the enforcement phases. Depending on the actual implementations of the PDPs and PEPs, this is a general scheme which addresses various functional architectures, including certificate authorities, resource managers, distributed services (such as grid middleware), and operating systems. This scheme was previously examined in the context of distributed applications. In this case, PDPs are application independent functions which,

providing the policy is general enough, are able to make decisions for any type of application; by contrast, PEPs are application dependent functions, which should be located in the same operating environment of the target [5].

2.3 Implementation of Security Service Environment

With the wide use of Internet-based applications, security in distributed systems becomes a serious issue to individuals, companies and organizations. This makes computer security a necessity to all computer users. Thus, many users use a security service that can be used to protect them from others. However, security services require extra computing resources because they are time-consuming. Normally to implement the authorization framework we require multi-agent system that acts as a middleware between the user and the network-centered computing environment. The main design principles of the system are that the system should provide the user with fast, efficient, stable and flexible security service. The agents interact cooperatively in a distributed environment and collectively act on behalf of the users to provide them with a secure environment [1, 2].

The accurate authentication of users is one of the main important issues in distributed information systems. An efficient authentication can accurately verify the identity of a user and then allows the user to access some pre-defined resources, such as reading a file or executing an application. Usually an individual can be authenticated by his/her knowledge of a system or his/her physical characteristics. An assigned password is a traditional way for an individual to be authenticated by showing his/her knowledge of the system. In an agent-oriented model, the security service is described in terms of two main types of entities: users and software agents, as shown in Figure 2.3. Agents work together cooperatively to satisfy the users' needs. Users are classified into two types: native and foreign users. Native users can access security services within the domain while foreign users can access the security service only through a proxy. Different users with different privileges can access different security services. This raises the following problems[1]:

- How to authenticate a user;
- How to grant the user certain permissions according to the domain and identity information.
- How to provide the user with fast and efficient security services (encryption, decryption, digital signature, verification, etc.)

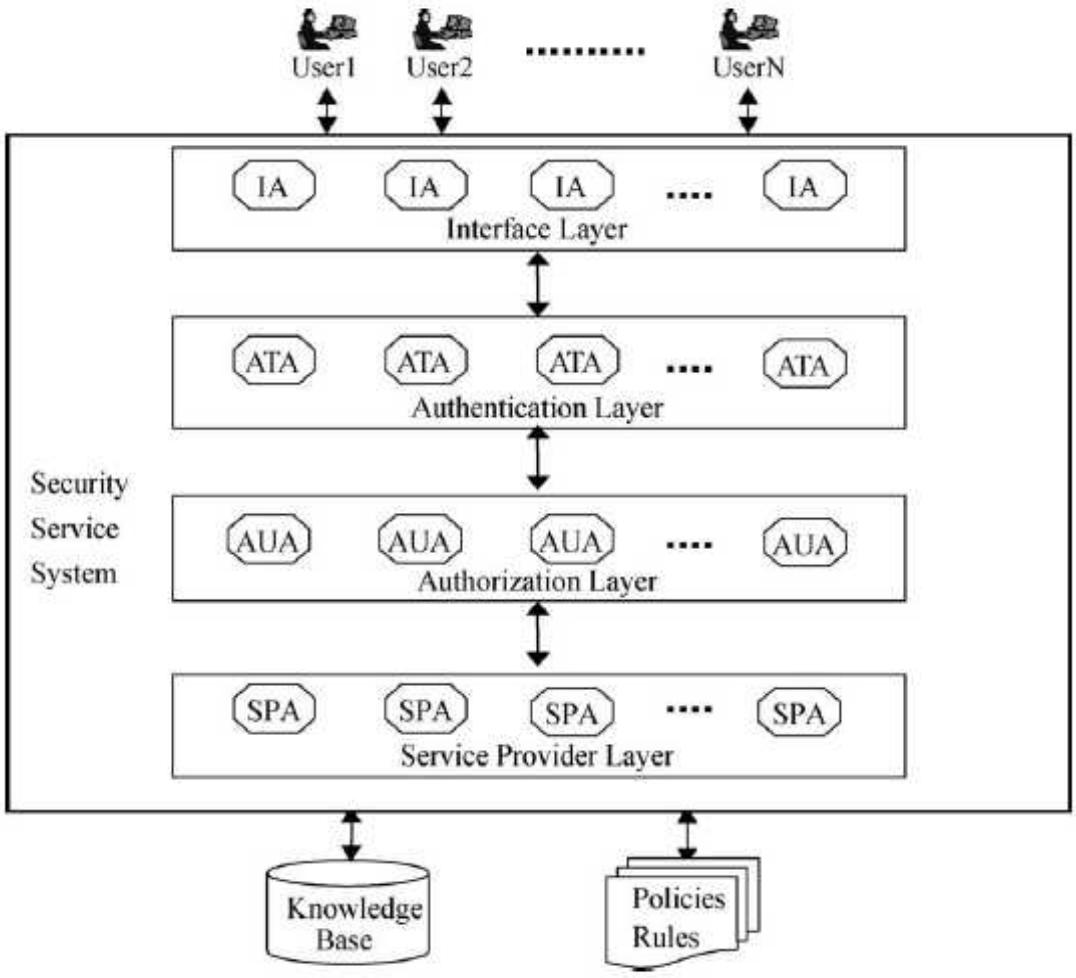

Figure 2.3:- Security service environment

(Source: E.Shakshuki et al. 2005)

2.4 Elements of the security service environment

2.4.1 Interface agents

Interface agents (IAs) are the users' intelligent interface to the system and allow the users to interact with the security service environment. The IA provides graphical interfaces to the user for interactions between the system and the user. IAs has the following actions [1]:

- Receive real-time keystroke from its user and sends them to the ATA.
- Receive login information from the user, such as: domain type, username, password and sends them to ATA.
- Receive success/fail login results from ATA and displays the results to the user.
- Receive service requests from a valid user and sends them to the AUA.
- Receive negotiation results from the AUA and displays them to the user.
- Receive the service results from SPA and displays them to the user.

2.4.2 Authentication agents

The ATA's main responsibility is to verify the identity of the user, using recognition of keystroke dynamics. It analyses the input information received from the IA and decides whether to accept or reject the current user (Figure 2.4.2). ATAs have the following actions [1]:

- Receive keystroke training data and user's login information.
- Train the neural network via user real-time keystroke data.
- Match the user's login keystroke with pre-loaded weights in the neural network.
- Send login results to the IA.
- Send valid login information to the AUA.
- Receive authorization results and forward them to the IA.

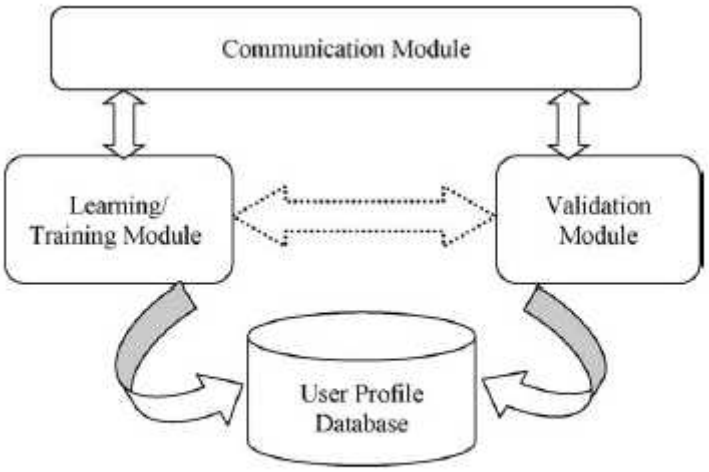

Figure 2.4.2:- Authentication agent architecture
(Source: E.Shakshuki et al. 2005)

2.4.3 Authorization agents

Authorization agents (AUAs) work at the third layer of the system and they do not have the capability to interact with the users directly. The authorization layer comprises three kinds of agents [1], as shown in Figure 2.4.3.

- Native-authorization agents (NAAs) are responsible for receiving authenticated native request messages from the ATAs. The NAA checks the access control database to authorize this native request. If it is an entitled request, a message is forwarded to the SPA to fulfill the desired service.
- Foreign-authorization-agents (FAAs) are responsible for receiving authenticated foreign request messages from the ATA. The FAA validates those requests by checking the foreign policy. If the negotiation succeeds, the FAA informs the SPA to conduct services in this specific level. Otherwise, the FAA sends the results back to the ATA.
- Foreign-delegation-agents (FDA) are responsible for delegating foreign users to get the highest level of service quality through negotiation with the FAA. A simple negotiation strategy is applied where the agent tries to gain credits from

foreign domains by serving foreign users, under the condition that the native requests are satisfied first.

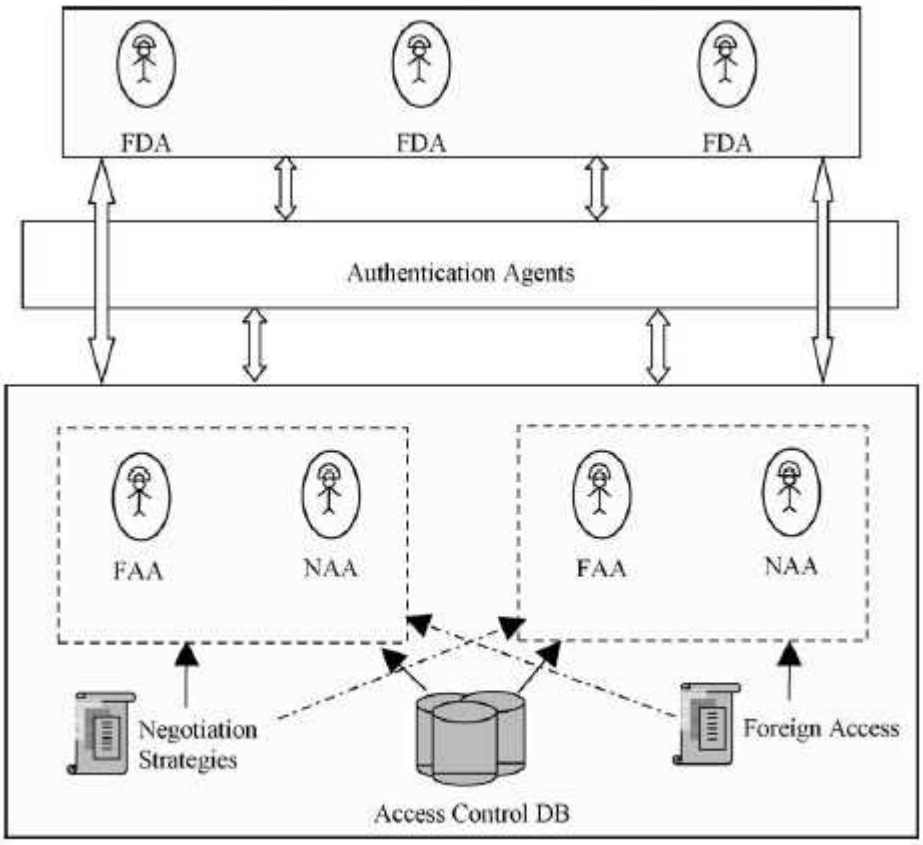

Figure 2.4.3:- Authorization layer

(Source: E.Shakshuki et al. 2005)

2.5 Authorization Framework Elements

An authorization system can be defined as a system that grants specific type of access to specific requesters based on their authentication, what services/resources they are accessing, current state of the system and their conformation to established policies. It is a detailed description of all aspects of a system that relate to access of services/resources by requesters. In order to understand the architecture well, some of the elements have identified and defined as follows [2]:

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

- Service (SR): Service is a piece of software that provides some functionality and can be accessed by Subjects or other Services. Services are exposed in the environment and are found by Subjects.
- Resource (R): Resource is an object that is accessed by Subjects. It can be a CPU, a storage device, software, data, scientific instrument or any other peripheral. Subjects access Resources through Services. In other words, a Resource is a Service.
- Service Policy (SP): Service Policy refers to the set of rules/requirements associated with the Service. A Subject must conform to Service Policy in order to Access that Service.
- Domain (DO): Domain refers to the set of Subjects and Services under a unique Domain Policy (DP): DO = (SU, SR, DP). The Services in a Domain are provided by Service Providers and they may belong to same or different physical organizations/institutions.
- Domain Policy (DP): Domain Policy refers to the set of rules/regulations/requirements of a Domain to which an entity must conform to in order to be in that Domain.
- Access (AC): Access is an operation that a Subject performs on Service/Resource. The access is provided based on conformance to Service Policy (SP) that is associated with that Service/Resource. Access operation is represented as AC (SU, SR, SP) i.e. Subject (SU) can access (AC) Service (SR) if it conforms to Service's Policy (SP).
- Policy (PO): Policy is a set of rules/requirements that can be associated with a Subject/Service/Domain. It can be represented as PO = (AP, AuP, TP, PP, MP, OP) where:

  AP is Authentication Policy.

  AuP is Authorization Policy.

  TP is Trust Policy.

  PP is Privacy Policy.

  MP is Management Policy.

- Domain Set (DS): Domain Set is simply the set of Domains.
- Domain Set Policy (DSP): Domain Set Policy refers to the Policy that applies to Domain Set.
- Filter: The rights/privileges of a Subject are different in different Domains. Filter is a component through which rights/privileges of a Subject are filtered for a particular Domain. There are two types of filters (filter-in and filter-out).
- MAP (MAP): is an operation that maps/transforms Subject of one administrative domain to Subject in another administrative domain. E.g. Subject-I (Domain-2) map Subject-J (Domain-l) means that subject Subject -I of domain Domain-2 has been mapped to subject Subject-J in Domain Domain-l.

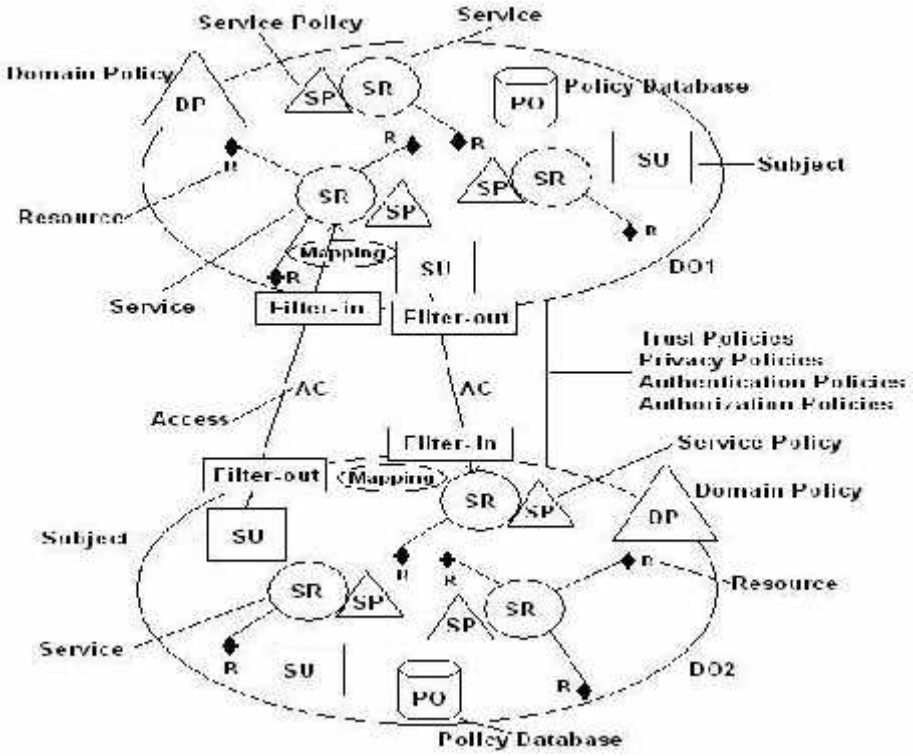

Figure 2.5:- Distributed Environment with two domains along with framework elements

(Source Singh,S. and S. Bawa 2007)

As shown in figure 2.5, When a subject request for service SR, request first passes through Filter-out component at the source Domain and then through Filter-in component

at the target Domain. During this passage, Subject's access rights are filtered for the target Domain. At the target Domain, the proposed 'Authorization Framework' checks Subject's conformance to Service Policy. If Subject conforms to Service Policy then mapping operation (MAP) is performed to provide the Subject an identity that is local to that Domain. The mapped identity is then used by the Target Domain to provide access to the requested Service/Resource. The mapping operation is optional. If the access of a Service/Resource does not require local identity then it can be omitted. If Subject does not conform to Service Policy, the access is denied. Determining whether a Subject conforms to Service Policy is a complex task and is performed by authorization engine which must take into account different types of policies like Trust Policies, Privacy Policies, Authorization Policies, and Authentication Policies etc. to grant/deny access to Services/Resources.

2.6 Access control

Access controls are security features that control how users and systems communicate and interact with other systems and resources. Access is the flow of information between a subject and an object.

- A subject is an active entity that requests access to an object or the data within an object. Eg. user, program, process etc.
- An object is a passive entity that contains the information. Eg. Computer, Database, File, Program etc.

Access controls give organizations the ability to control, restrict, monitor, and protect resource availability, integrity and confidentiality.

2.6.1 Access Control Technologies

Single Sign-On:

SSO is a technology that allows a user to enter credentials one time and be able to access all resources in primary and secondary network domains [10].

Advantages:

- Reduces the amount of time users spend authenticating to resources.
- Enable the administrator to streamline user accounts and better control access rights.
- Improves security by reducing the probability that users will write down their passwords.
- Reduces the administrator's time in managing the access permissions.

Limitations:

Every platform application and resource needs to accept the same type of credentials, in the same format and interpret their meaning in the same way.

Disadvantages:

Once an individual is in, he is in, thus giving a bigger scope to an attacker.

Kerberos [10]:

- Kerberos is an authentication protocol that was designed in mid 1980 as part of MIT's project Athena.
- It works in a C/S model and is based on symmetric key cryptography
- It is widely used in UNIX systems and also the default authentication method for windows 2k and 2k3 and is the de-facto standard for heterogeneous networks.

Kerberos Components

- Key Distribution Center (KDC)
  - Holds all users and services secret key and info about the principles in the database
  - Provides an authentication service with the help of a service called AS
  - Provides key distribution functionality
  - Provides a ticket granting service (TGS)

- Secret Keys are the keys shared between principle and KDC generally using symmetric key cryptography algorithm that are used to authenticate the principles and communicate securely
- Principles are users, applications or any network services
- A ticket is a token generated by KDC and given to a principle when one principle need to authenticate another principle
- Realm is a set of principles. A KDC can be responsible for one or more realms. Realms allow an administrator to logically group resources and users.
- Session Keys are the keys shared between the principles that will enable them communicate security.

Kerberos Authentication Process

- User enters username and password into the workstation (WS)
- The Kerberos s/w on the workstation sends the username to the Authentication Server (AS) on the KDC.
- The AS generates a Ticket Granting Ticket (TGT) encrypting it with the user's secret key stored in DB with the help of TGT and sends it to the user.
- The password entered by the user is transformed into a secret key using which the ticket (TGT) is decrypted and thus the user gains access to the WS.
- Suppose the user wants to use the printer, the users system send the TGT to the TGS on the KDC
- The TGS generates a new ticket with two instances of a session key, one encrypted with the user's secret key and the other encrypted with the print server's secret key. This ticket may also contain an authenticator which contains info on user.
- The new ticket is sent to the users system which is used to authenticate with the print server.
- The user's system decrypts and extracts the session key, adds a second authenticator set of identification information to the ticket and sends the ticket onto the print server.

- The print server receives the ticket, decrypts and extracts the session key, and decrypts and extracts the two authenticators in the ticket. If the printer server can decrypt and extract the session key, it knows that the KDC created the ticket, because only the KDC has the secret key that was used to encrypt the session key. If the authenticator information that the KDC and the user put into the ticket matches, then the print server knows that it received the ticket from the correct principal.

Weakness of Kerberos

- The KDC can be a single point of failure. If the KDC goes down, no one can access needed resources. Redundancy is necessary for the KDC.
- The KDC must be able to handle the number of requests it receives in a timely manner. It must be scalable.
- Secret keys are temporarily stored on the users' workstation, which means it is possible for an intruder to obtain these cryptographic keys.
- Session keys are decrypted and reside on the users' workstations, either in a cache or in a key table. Again, an intruder can capture these keys.
- Kerberos is vulnerable to password guessing. The KDC does not know if a dictionary attack is taking place.
- Network traffic is not protected by Kerberos if encryption is not enabled.

SESAME[10]:

- SESAME (Secure European Systems for Applications in a Multi-vendor Environment) is a SSO technology that was developed to extend Kerberos functionality and improve upon its weakness.
- SESAME uses a symmetric and asymmetric cryptographic technique to protect exchanges of data and to authenticate subjects to network resources.

- SESAME uses digitally signed privileged Attribute Certificates (PAC) to authenticate subjects to objects. PAC contains the subject's identity, access capabilities for the object, access time period, and life time of the PAC.

Security Domain:

- A domain is a set of resources that are available to a subject.
- A security domain refers to the set the resources working under the same security policy and managed by the same group.
- Domains can be separated by logical boundaries, such as
    - Firewalls with ACL's
    - Directory services making access decisions
    - Objects that have their own ACL's indicating which individual or group can access them.
- Domains can be architected in a hierarchical manner that dictates the relationship between the different domains and the ways in which subjects within the different domains can communicate.
- Subjects can access resources in domains of equal or lower trust levels.

Thin Clients:

- Thin clients are diskless computers that are sometimes called as dumb terminals.
- It is based on C/S technology where a user is supposed to logon to a remote server to use the computing and network resources.
- When the user starts the client, it runs a short list of instructions and then points itself to a server that will actually download the operating system, or interactive operating software, to the terminal. This enforces a strict type of access control, because the computer cannot do anything on its own until it authenticates to a centralized server, and then the server gives the computer its operating system, profile, and functionality.

- Thin-client technology provides another type of SSO access for users, because users authenticate only to the central server or mainframe, which then provides them access to all authorized and necessary resources.

2.6.2 Access Control Models

An access control model is a framework that dictates how subjects access objects. It uses access control technologies and security mechanisms to enforce the rules and objectives of the model. There are three main types of access control models [5]:

- Discretionary.
- Mandatory
- Nondiscretionary (also called role-based).

Discretionary Access Control

- The control of access is based on the discretion (wish) of the owner
- A system that uses DAC enables the owner of the resource to specify which subjects can access specific resources
- The most common implementation of DAC is through ACL's which are dictated and set by the owners and enforced by the OS.
- Examples: Unix, Linux, Windows access control is based on DAC
- DAC systems grant or deny access based on the identity of the subject. The identity can be user identity or a group identity (Identity based access control)

Mandatory Access Control

- This model is very structured and strict and is based on a security label (also known as sensitivity label) attached to all objects
- The subjects are given security clearance by classifying the subjects as secret, top secret, confidential etc) and the objects are also classified similarly
- The clearance and the classification data is stored in the security labels, which are bound to the specific subject and object.

- When the system makes a decision about fulfilling a request to access an object it is based on the clearance of the subject. The classification of the object and the security policy of the system
- This model is used and is suitable for military systems where classifications and confidentiality is of at most important
- SE Linux, by NSA, trusted Solaris are examples of this model
- Security label are made up of a classification and categories, where classification indicates the security level and the categories enforce need to know rules.

Non Discretionary or Role Based Access Control

- A RBAC is based on user roles and uses a centrally administered set of controls to determine how subjects and objects interact.
- The RBAC approach simplifies the access control administration
- It is a best system for a company that has high employee turnover.
- Note: The RBAC can be generally used in combination with MAC and DAC systems.

2.6.3 Access Control Techniques

Different access control technologies are available to support the different access control models [3, 6].

- Rule-Based Access Control
- Constrained User Interface
- Access Control Matrix
- Content Dependant Access Control
- Context Dependant Access Control

Rule-Based Access Control

- Rule-based access control uses specific rules that indicate what can and cannot happen between a subject and an object.
- A subject should meet a set of predefined rules before it can access an object.

- It is not necessarily an identity based i.e. it can be applicable to all the users or subjects irrespective of their identities.
- Eg. Routers and firewall use rules to filter incoming and outgoing packets

Constrained User Interface

- Constrained user interfaces restrict user's access ability by not allowing them to request certain functions or information, or to have access to specific system resources.
- There are three major types of restricted interfaces:
    - Menus and Shells:
    - Database Views
    - Physically Constrained Interfaces.

Access Control Matrix

- An access control matrix is a table of subjects and objects indicating what actions individual subjects can take upon individual objects.
- The access rights that are assigned to individual subjects are called capabilities and that assigned to objects are called Access Control Lists (ACL).
- This technique uses a capability table to specify the capabilities of a subject pertaining to specific objects. A capability can be in the form of a token, ticket, or key.
    - Each row is a capability and each column is an ACL for a given user.
    - Kerberos uses a capability based system where every user is given a ticket, which is his capability table.
- ACL's are list of subjects that are authorized to access a specific object and they define what level of authorization is granted (both at individual and at group level)
- ACL's map values from the access control matrix to the object.
- Note: A capability table is bound to a subject, whereas an ACL is bound to an object.

Content Dependant Access Control

- Access to the objects is based on the content within the object.
- Example: Database Views, E-mail filtering etc.

Context Dependant Access Control

The access decisions are based on the context of a collection of information rather than on the sensitivity of the data. For example: A firewall makes a context-based access decisions when they collect state information on a packet before allowing it into the network.

2.7 Authorization Process

Authorization is the mechanism by which a system determines what level of access a particular authenticated user should have to secure resources controlled by the system. Authorization systems depend on secure authentication systems to ensure that users are who they claim to be and thus prevent unauthorized users from gaining access to secured resources. An authorization mechanism is responsible for conducting the communication required to become authorized to join a group. Subject's access request for Service SR first passes through Filter-out component at the source Domain and then through Filter-in component at the target Domain. During this passage, Subject's access rights are filtered for the target Domain. At the target Domain, the proposed Authorization Framework checks Subject's conformance to Service Policy. If Subject conforms to Service Policy then mapping operation (MAP) is performed to provide the Subject an identity that is local to that Domain. The mapped identity is then used by the Target Domain to provide access to the requested Service/Resource. The mapping operation is optional. If the access of a Service/Resource does not require local identity then it can be omitted. If Subject does not conform to Service Policy, the access is denied. Determining whether a Subject conforms to Service Policy is a complex task and is performed by authorization engine which must take into account different types of policies like which must take into account different types of policies like, Authentication Policies etc. to grant/deny access

to Services/Resources [1, 2].

There is a Policy database to store different types of policies. These Policies exist in a complex manner among Subjects and Services of different Domains. These models can be used to provide privacy, trust and policy based access to Services/Resources. Authorization request from Subject SU is first intercepted by PEP (Policy Enforcement Point). PEP constructs an authorization decision query and passes it to authorization handler. The result of this query determines whether the access to Service/Resource is granted. The authorization decision query has details about the identity of the Subjects and the Service requested. Authorization Engine passes this information to PDP (Policy Decision Point). The Policy is retrieved by PDP from PRP (Policy Retrieval Point). If the policy information is not available at PRP, it may be retrieved from Policy Store.
The Policies are written by administrator using PAP (Policy Administration Point). The Policy Store is capable of importing/exporting XACML. In XACML, Policy is constructed as a set of rules against the target defined as a triad (Subject, Resource, and Action). PIP (Policy Information Point) is used by Authorization Engine to retrieve Resource, Subject and Environment related attributes. Trust Manager and Privacy Handler provide trust and privacy based access information to Authorization Engine. After getting this information, Authorization Engine prepares a final result and passes it to PEP. Based on the result, PEP then grants/denies access to the Service/Resource. Obligation Service, if any, is also executed by PEP [1, 2].

The authorization framework may uses a list of Policy Information to gather attributes about entities and a chain of Policy Decisions to evaluate whether a said action can be performed by an entity on the resource.

Authorization Requirements [1, 2]

- Resource managers at the target site must be able to define policy rules which specify the set of entitlements required to obtain access to a particular web resource.
- A properly configured web server must ensure that only those requestors having

the appropriate entitlements can access the resource(s).

- The target site must manage authorization, matching an entitlement provided by the origin site against the appropriate policy rules.
- Resource owners should be able to specify in a policy rule that "group X at site Y" can access the resource. Shibboleth should be able to express this idea.

Pre-requirements for Authorization [1, 2]

- Each origin site must have an existing local area wide authentication system.
- Users must authenticate to web services using the method their related local-area has documented for them. User should never be asked to type their passwords into a remote authentication service.
- Policy issues (such as "how long does a login last") are considered to be local issues, out of scope for Shibboleth.
- There must be a standard way for the local authentication system to assert the local identity of the browser user to the local Attribute Authority. A user MUST be able to access remote resources without the target site implementing a version of the origin site's authentication system.
- If the intra-institutional user experience is single sign-on (i.e., "login", however that happens, only once for many/all local sites), then the inter-institutional access should also have that same experience. However, only-once-for-all-external-accesses (in addition to a local sign-on) would probably be acceptable.

Requirements of Browser User Environment [6]

- Shibboleth must work with common-off-the-shelf browsers (i.e. the "current" version of the Microsoft and Netscape browsers, with current defined by the marketplace).
- Shibboleth should strive to avoid requiring Shibboleth-specific software on client machines. This should be done only when there is absolutely no alternative.
- The Shibboleth design MUST not require usable client certificates.

Requirements of Web Server Environment [6]

- The implementation must run with the Apache web server.
- A site must be able to participate in Shibboleth as an origin or as a target whether or not they are running a local SSO implementation.
- Once access control has completed, session management must be handled by the web application.

CHAPTER 3

METHODOLOGY

The implementation of authorization framework is totally based on a secure communication system. The authorization framework is implemented on a Public Key Infrastructure (PKI) which provides the authentication of the user in a securely and transparently on an insecure communication media that is internet. For this purpose at first, the literature review and the analysis of the PKI-infrastructure is performed. For the successful implementation of authorization framework it will take a lot of time.

There is one automated authentication and authorization tool called shibboleth with the help of this tool, this work can be done in a short duration of time. The shibboleth is one of the most widely used authorization tool used to allow the users to access the resources placed online based on the authentication level. All the required information about shibboleth tool is studied from various research papers and related websites. The issue of authorization comes only after the authentication process so it based on the authenticated system. For this purpose at first, the authentication, its requirements, processes, and various authentication technologies are studied and analyzed. Then requirements and needs for the authorization system and various policies and mechanisms of the authorization are studied and analyzed.

Since the authorization tool, shibboleth will be used to implement the authorization framework, the detailed study of how it works and its infrastructure is needed to be performed. To achieve the objectives of the research project, its implementation, its strengths and limitations need to be reviewed in detail. It will be able to implement the authorization framework to make it work in the real world only after these process and methodologies.

Everything done over here is a theoretical research of authentication and authorization process. To understand how authentication and authorization can be implemented in real world application, an application will be demonstrated. The application will give the

concept of how the resources are protected from unauthorized access and it is developed using Progress and X/E Files.

3.1 Automated Authentication and Authorization Tool (Shibboleth)

Shibboleth, a joint project of Internet2/MACE and IBM, is investigating architectures, frameworks, and practical technologies to support inter-institutional sharing and controlled access to web available services.

Shibboleth is a system designed to exchange information across realms for authentication and authorization. It provides a secure framework for one organization to transmit attributes about a web-browsing individual across security domains to another institution. In the primary usage case, when a user attempts to access a resource at a remote domain, the user's own home security domain can send certain information about that user to the service provider site in a trusted exchange. These attributes can then be used by the resource to help determine whether to grant the user access to the resource [7].

The user may have the ability to decide whether to release specific attributes to certain sites by specifying personal Attribute Release Policies (ARP's), effectively preserving privacy while still granting access based on trusted information. A service provider (SP) protects resources, while an identity provider (IdP) provides information about users from an organization to service providers. When a user first tries to access a resource protected by a Shibboleth SP, they are redirected to a service which asks the user to specify the organization from which they want to authenticate. If the user has not yet locally authenticated for that IdP, the user will then be asked to authenticate using any mechanism of that site's choosing. After the user authenticates, the Shibboleth IdP at the local institution generates a temporary reference to the user, known as a handle, for the individual and sends this to the SP [7, 8].

The SP can then use the handle to ask for attributes about this individual. Based on these attributes, the SP can decide whether or not to grant access to the resource. The user may

then be allowed to access the requested materials. The key benefits of using Shibboleth are:

- Reduces time needed to manage access to protected resources.
- Increases security
- Interoperates with similar standards-based solutions

Shibboleth is one of the most widely used authorization tool which provide:

- Relief from multiple passwords and sign-on, along with the resulting improvements in security.
- Protection against unnecessary disclosure of personal attributes, resulting in preservation of privacy.

There are several controls on privacy in Shibboleth, and mechanisms are provided to allow users to determine exactly which information about them is released. A user's actual identity isn't necessary for many access control decisions. Instead, the resource often utilizes other attributes such as faculty member or member of a certain class. While these are commonly determined using the identity of the user, Shibboleth provides a way to mutually refer to the same principal without revealing that principal's identity. Because the user is initially known to the SP site only by a randomly generated temporary handle, if sufficient, the SP site might know no more about the user than that the user is a member of the IdP organization. This handle should never be used to decide whether or not to grant access, and is intended only as a temporary reference for requesting attributes [7, 8].

3.1.1 Identity Provider (IDP)

There are four primary components to the IDP component in Shibboleth: the Attribute Authority (AA), the Handle Service (HS), attribute sources, and the local sign-on system (SSO). The AA and HS are provided with Shibboleth. Any single sign-on service that's capable of providing REMOTE_USER may be used with Shibboleth. The attribute sources are provided by the surrounding infrastructure in the form of databases or

directories. From the identity provider site's point of view, the first contact will be the redirection of a user to the handle service, which will then consult the SSO handler to determine whether the user has already been authenticated. If not, then the browser user will be asked to authenticate, and then sent back to the assertion consumer service URL with a handle bundled in an attribute assertion. Next, a request from the Shibboleth daemon, shibd, will arrive at the attribute query handler which will include the previously mentioned handle. The IDP then consults the ARP's for the directory entry corresponding to the handle, queries the directory for these attributes, and releases to shibd all attributes the service provider is entitled to know about that user [7].

3.1.2 Service Provider (SP)

There are three primary components to the SP component in Shibboleth: the Assertion Consumer Service (ACS), the Attribute Requester (AR), and the resource manager (RM). An implementation of each of these is included in the standard Shibboleth distribution. These components are intended to run on the same web server. From the service provider's point of view, a browser initially makes a request for a Shibboleth-protected resource. The resource manager allows the service provider to step in, which will use the WAYF to acquire the name of an identity provider to ask about the user. The IdP will then reply with a SAML authentication assertion containing a handle, which the assertion consumer service then hands off to shibd. Shibd uses the handle and the supplied address of the corresponding attribute query handler to request all attributes it is allowed to know about the handle. The resource manager performs some basic validation and analysis based on attribute acceptance policies (AAP's). These attributes are then handed off to the application or used internally to decide whether to grant access [7].

3.1.3 Where are you from? (WAYF)

The WAYF service can be either outsourced and operated by a federation or deployed as a part of the ACS. It is responsible for allowing a user to associate themselves with an institution of their specification, then redirecting the user to the known address for the handle service of that institution [7, 8].

3.1.4 Federations

A Shibboleth federation provides part of the underlying trust required for function of the Shibboleth architecture. A federation is a group of organizations (universities, corporations, content providers, etc.) who agree to exchange attributes using the SAML/Shibboleth protocols and abide by a common set of policies and practices.

3.1.5 Relying Parties

Some aspects of both IdP and SP configuration can vary and be expressed in terms of the "relying party". To an IdP, an SP is a relying party, while SPs consider IdP's to be relying parties. Certificates, policies, and other aspects of an interaction are specified on the basis of the relying party, and may or may not vary between relying parties depending on the deployment's needs.

3.1.6 Applications

Shibboleth "applications" are the primary unit of SP configuration. Applications as viewed by the SP implementation are not necessarily defined by the same metrics as in other contexts. An individual application represents a set of web resources that operates using the same attribute handling and trust configuration and shares a common session with the browser user. As a user navigates between resources on a server that cross an application boundary, a new session is established, though user interaction may not be required. As a consequence of the relationship between applications and sessions (which are tracked with a cookie), an application usually does not span more than one virtual host. Apart from cookie-based constraints, web resources can be aggregated into applications in arbitrary ways.

3.2 The working structure of Shibboleth system

When a user request (normally via a web browser) to access the resources managed by Shibboleth, it actually issue a request to Shibboleth Service Provider. Once this event is received by the Shibboleth SP, it will try to retrieve the user's identity from the Shibboleth Identity Provider using a service called Where Are You From (WAYF). If the user is known to the Shibboleth IdP, the user's attributes will be sent back to SP. Otherwise, the user will be asked to authenticate himself. The user has full control over the attributes he provides to the IdP using the Attribute Release Policies (ARP's). Figure 3.2 shows this structure [8].

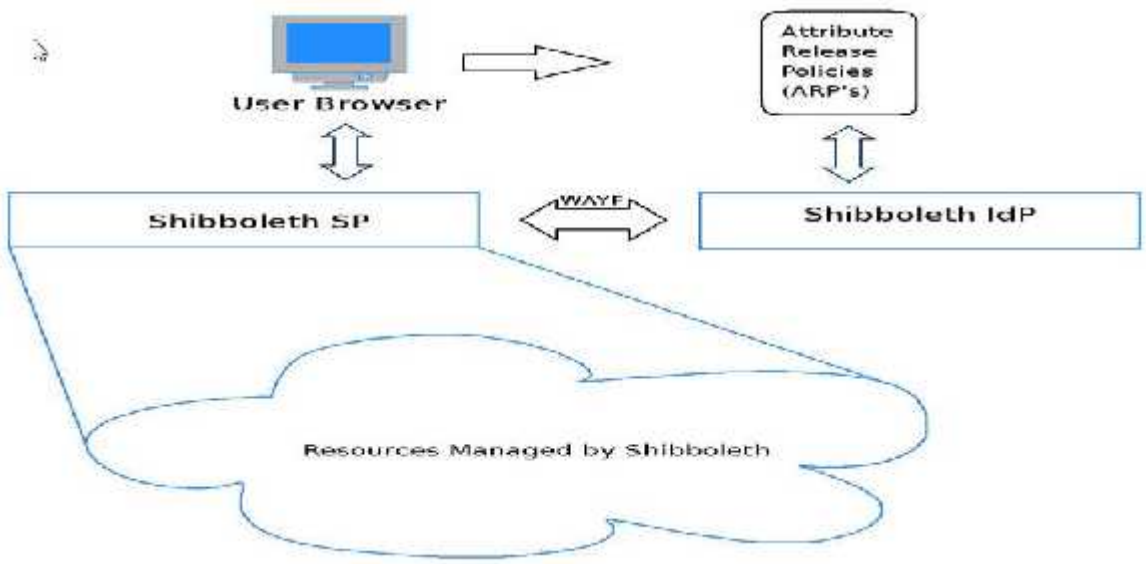

Figure 3.2:- The Structure of Shibboleth System.

(Source: Welch, Von, et al, 2005)

3.2.1 Authentication Procedure

What happens if a user requests the resources managed by Shibboleth? As soon as a user (web browser) issues a request for the resource, an event is triggered in Resource Manager. Resource manager will then issue a request to the SSO handler (of corresponding IDP) using the WAYF service. What the Idp does at this moment is to feedback the request with a handle using SAML. This handle has a URL about the

current user which can be used to locate the user's attributes. Assertion Consumer Services takes care of the returned handle by sending it to `shibd'. As a result, shibd is then able to request the user's attributes from the Attribute query handler of IdP. The Attribute query handler determines what attributes should be revealed to shibd according to the user-defined `Attribute Release Policies', and informs the Directory Service on IdP to send the attributes to shibd accordingly. Finally, the user's attributes are transferred back to the Resource Manager which will make a decision whether or not the user should be granted the access. Such decision is made based on the Attribute Acceptance Policies. The following Figure 3.2.1 depicts the whole procedure of authentication [8].

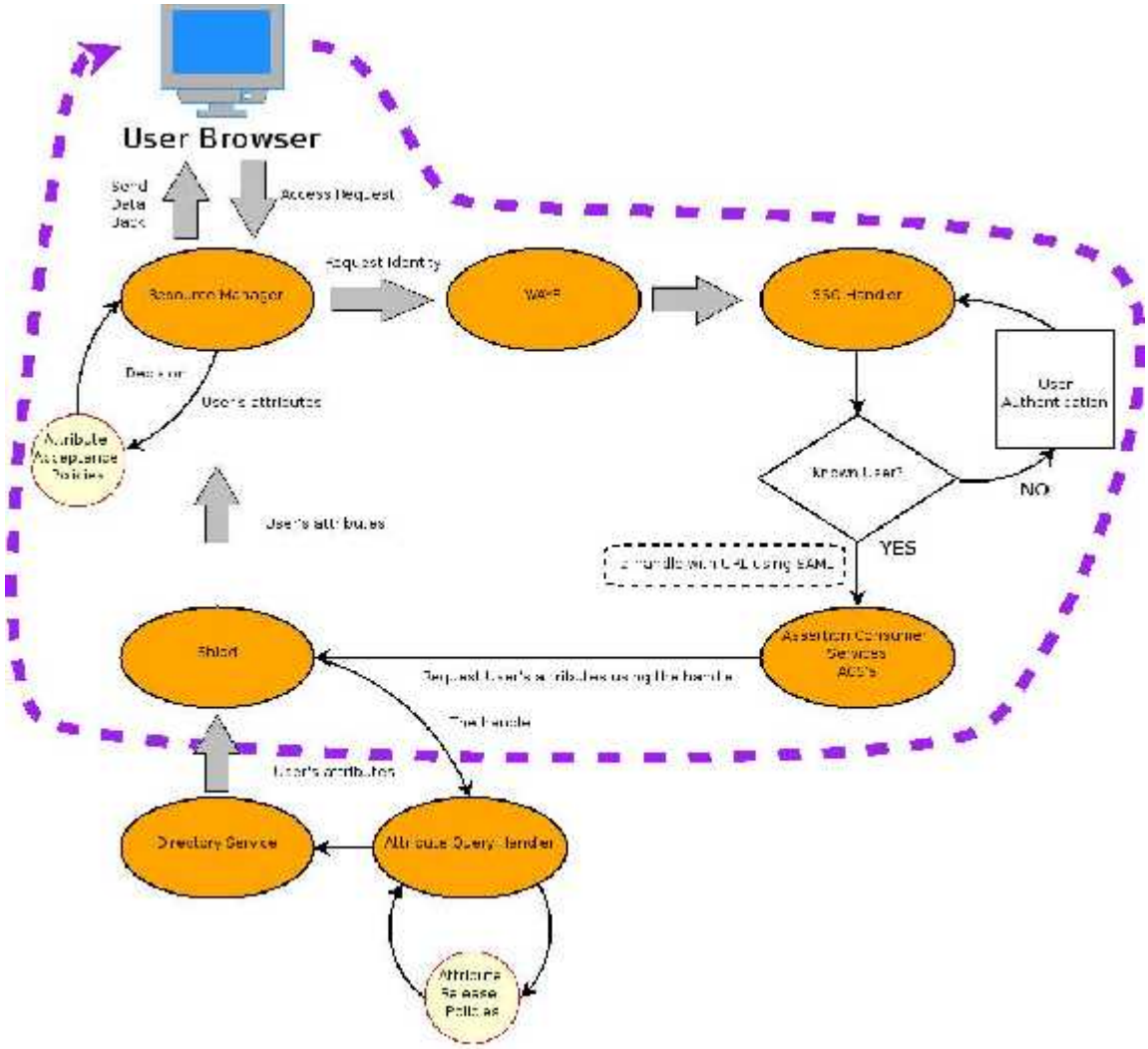

Figure 3.2.1:- Authentication procedure

(Source: Welch, Von, et al, 2005)

3.2.2 Shibboleth Flows

The Figure 3.2.2 below shows the flows during a "complete" Shibboleth-enabled transaction, with the browser user arriving at the Service Provider site without an existing session and without any information about their IdP site. There are many variations on this flow, most of them a lot simpler. However, this is offered as a starting point, for the sake of completeness [8].

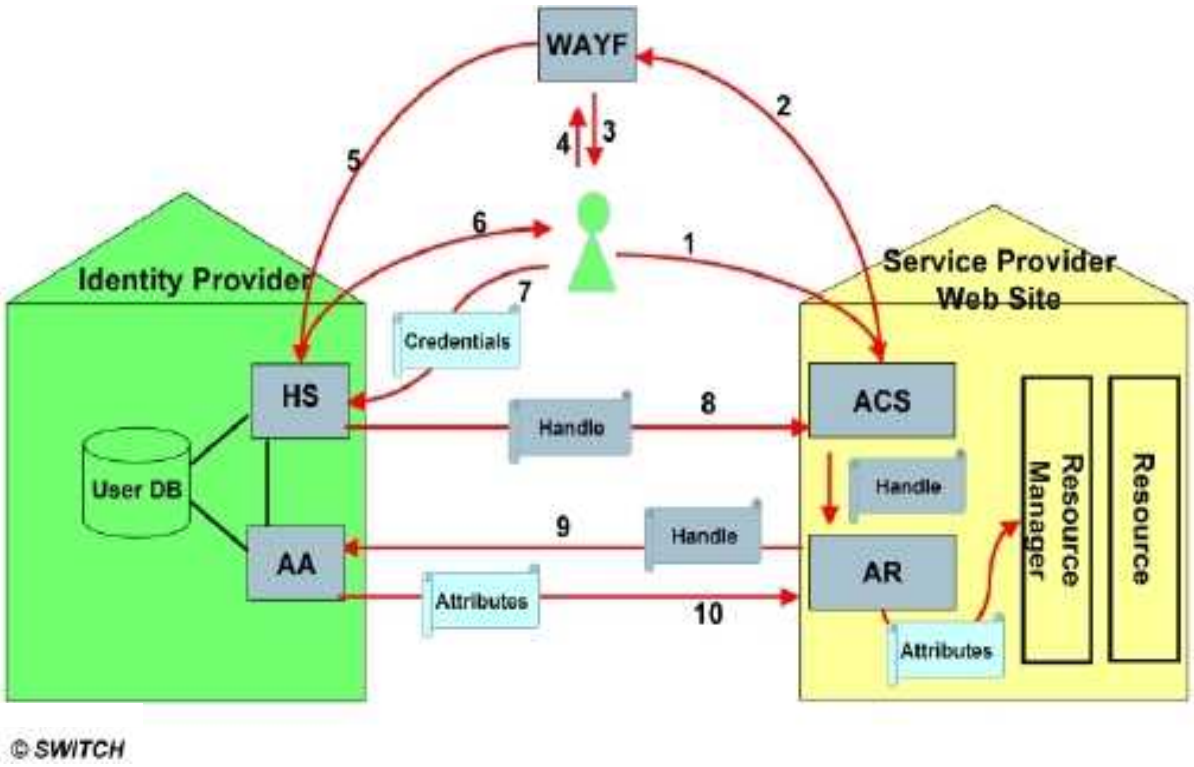

Figure 3.2.2: Shibboleth Flows

(Source: Welch, Von, et al, 2005)

1. User attempts to access Shibboleth-protected resource on SP site application server.

2, 3, 4. User is redirected to a Where Are You From (WAYF) server, where the user indicates their home site (IdP).

5. User is redirected to the Handle Service at their IDP.

6, 7 User authenticates at their IdP, using local credentials.

8. Handle service generates unique ID (Handle) and redirects user to Service Provider site's Assertion Consumer Service (ACS). ACS validates the supplied assertion, creates a session, and transfers to Attribute Requestor (AR).
9, 10. AR uses the Handle to request attributes from the IdP site's Attribute Authority. The attribute authority responds with an attribute assertion subject to attribute release policies; SP site uses attributes for access control and other application-level decisions.

3.2.3 Advantages of using Shibboleth

Shibboleth offers a compelling alternative for providers of services and content to higher Ed, by eliminating the need to build extensive custom front-ends and interfaces to deal with the variety of systems customer sites use for controlling access to resources and services. Shibboleth is freely available open source software, available for Solaris, Linux, and Windows 2000/XP. The following is a brief high-level overview of some of the advantages Shibboleth offers[7, 8].

- Unified authentication mechanism from the vendor perspective, much more scalable and much less integration work required to bring to new customer online.
- Ability to implement fine-grained access control (e.g. access by role), allowing customer sites to effectively control access by attributes and thus control usage costs, by not granting access unnecessarily – compelling marketing message for vendor.
- Ability to market yourself as being at the forefront of compelling new technology adoption. E.g., as Shibboleth enables role based access control (RBAC); vendors are able to offer new service offerings.
- Once the initial Shibboleth integration work has been completed on the vendor’s systems, the incremental cost of adding new customers is relatively minimal, in contrast.

3.2.4 Requirements of Shibboleth Identity Provider (IDP)

- A common institutional directory service should be operational; Shibboleth comes with JNDI and JDBC capabilities built in, which encompasses SQL and LDAP, and the Attribute Authority has a Java API which will allow specification of custom connectors to other types of data sources.
- A method to authenticate browser users must be in place, preferably in the form of an enterprise authentication service. Some form of an SSO or a WebISO service is not explicitly necessary for Shibboleth; however, it is highly recommended.
- Shibboleth is known to work on Windows, Linux, Mac OS X, and Solaris, but should function on any platform that has a Tomcat implementation.
- A Java servlet container; Shibboleth has been tested extensively with Tomcat.
- It is recommended that a web server such as Apache be deployed in front of Tomcat to provide authentication services and to control the flow of requests to Tomcat. There may be issues surrounding the number of maximum connections to the web server and to the servlet container [8].

3.2.5 Requirements of Shibboleth Service Provider (SP)

An IIS or Apache webserver must be deployed which is capable of SSL and running Shibboleth [7].

- Web applications must be modified to be able to rely on attributes supplied by Shibboleth. Often this will overlap with the same header variables set by other authentication schemes, such as REMOTE_USER .
- Access control schemes can often be simplified and rewritten to take advantage of the inherent power of attribute-based protection.

Server Certs

In the Shibboleth architecture, the identity provider and service provider must each have

various client and/or server KeysAndCertificates for use in signing assertions and creating SSL channels. These should be issued by a commonly accepted CA, which may be stipulated by some Federation rules. Different federations may require the use of different CA's [7].

Attribute Release Policies

- The Attribute Authority maintains a set of policies called Attribute Release Policies (or ARP's) that govern the sharing of user attributes with Shibboleth service provider sites. When a user attempts to access a Shibboleth-protected resource, that resource's SP queries the user's IdP for all attributes to which it is entitled. shibd provides the URI of the SP and the URI of the requesting application. The attribute query handler finds the attributes associated with the browser user, determines an "Effective ARP" for this user, and then sends to the SP only the attribute-value pairs allowed in this policy.
- An ARP may be thought of as a sort of filter for outbound attributes; it cannot create attributes or data that aren't identity provider ally present, but it can limit the attributes released and the values those attributes may have when released. It does not change the information in the data sources in any way.
- Each ARP is comprised of one or more rules that specify which attributes and values may be released to a given application and that SP. The assignment of rules to various service providers is quite flexible and includes mechanisms for specifying: that a rule should affect all service providers (default rule), exact SP names for which a rule is applicable, regular expressions against which SP names should be matched to determine if a rule is applicable, and individual applications that may span hosts, URL's, or even SP's as necessary.
- Various ARP's may be combined in forming the Effective ARP. For instance, the Site ARP is administratively maintained and applies to all users for which the IdP is authoritative. User ARP's apply to a specific user only, and can be maintained either administratively or by the users themselves. All ARP's are specified using the same syntax and semantics [8].

CHAPTER 4

DESIGN AND IMPLEMENTATION

4.1 Concept Development

The concept of this model is to develop ‘service request management system’ which implements various authorizations for accessing a particular resource in this case service request access based on various roles which help users-service requesters (clients, staff, and managers) to deal efficiently with the service requests.

The proposed model can be used to post tasks and service requests by external and internal users. The system will be used as a management tool to manage and track the tasks and service requests. It will also work as a communication channel in between users related to tasks and service requests.

The various security aspects defined for this model is described below:

Company: This will be the top level of scope for user’s portfolio. There will be one and only one Host Company, and others will be the Non-host Company (customers).

Department: It is presumed that the system will have at least one department underneath of each company. This will be 2nd level of scope for user’s portfolio. This will be useful to look into the effort or involvement made by certain department in any project.

4.2 Implemented Roles in System

There will be two types of roles:

- Application Role
- Project Role.

These Roles will have system level, company level and project level of access scope. The scope will limit access and define visibility for the users. Like creating, Updating, Deleting and viewing company, departments, projects and service requests.

Each user will have access to the Application determining user's privileges from assigned Application Roles and Project Roles to her/him.

Following are assumed built in User Groups.

Application Roles: System Admin, Company Admin, User

Project Roles: Project Leader, Project Member.

System Admin will have access to the System level functions whereas Company Admin will have access to the Company level functions. Rest of groups will have Project level access with varying set of permission inside the project.

Role Permission:

Role Permission will be of have two types.

- Application Role Permission
- Project Role Permission.

The User Roles will be privileged for access to the different level of application/project access. A user will access privileges through assigned roles to her/him. All users will be created underneath of Departments. Users will be privileged with Application and Project Roles. According to the Roles assigned they will have access scope either to system level, company level or project level. By default every user will have normal-user privileges. User having privileges to the User Role and from Host Company will be defined as **Internal User or Host Company User,** but user from Customer Company will be defined as **External Company User or Non Host Company User.**

User will have three levels of access scopes:

- System level scope
- Company level scope
- Project level scope.

This will be based on which user role they have been privileged to.

Project

A company may have many projects. All projects will be hiding under the scope of company for the External Company User. Every user will have access to the projects of their own company only. The external company user only can see the projects of their own company but host company user may have access to the project of other companies through project access role permission. Project visibility is granted as per the project permissions & user role association.

Project Team

Project team is a composition of users assigned with certain user roles to act on certain list of projects. To be member of Project Team a user must be privileged with role Project Leader or Project Member. The number of role provided to the specific user for specific project will not limit but the permission privileged will be the union of all user role privileges assigned.

Service Request (SR)

A project will have many SRs. A SR may be for Bug, Task, Feature Request, Change Request, or Support SR. SRs will be either internal or external. Being a SR internal mean that these SR will be visible for Host Company Users only. SRs posted by a user belonging to the host company will be default by Internal. However, at the time of creating a SR the user from Host Company will have option to mark the SR as external. This permission is given to host company users only. SRs posted by users in a non-host company will always be external. The Accessibility of SRs under internal/external view will be as defined below:

Internal: Accessible for internal users only.

External: Accessible for every one (Internal and External)

SRs will have internal & external view. This means that the things presented to the users of other company then the host company will have different view. SRs will hide under

the scope of Projects and will be accessible through role permission. Accessibility to the different part of SR in different phase and mode will vary for different user roles.

The overall system model and various application units and their relationships is diagrammatically described in figure below.

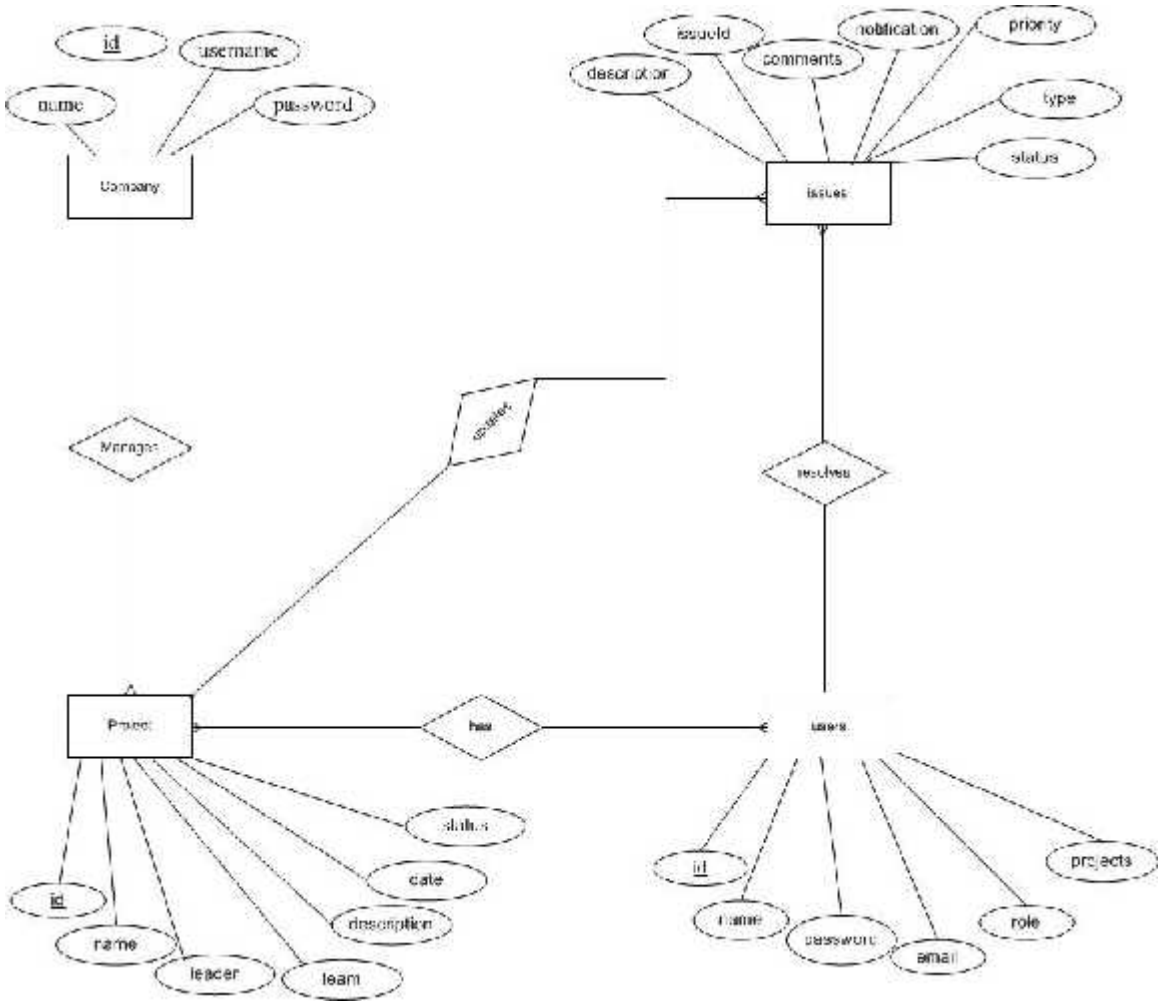

Figure 4.2:- Entity Relationship Diagram of the proposed model

4.3 Flowcharts for Different Role Permissions

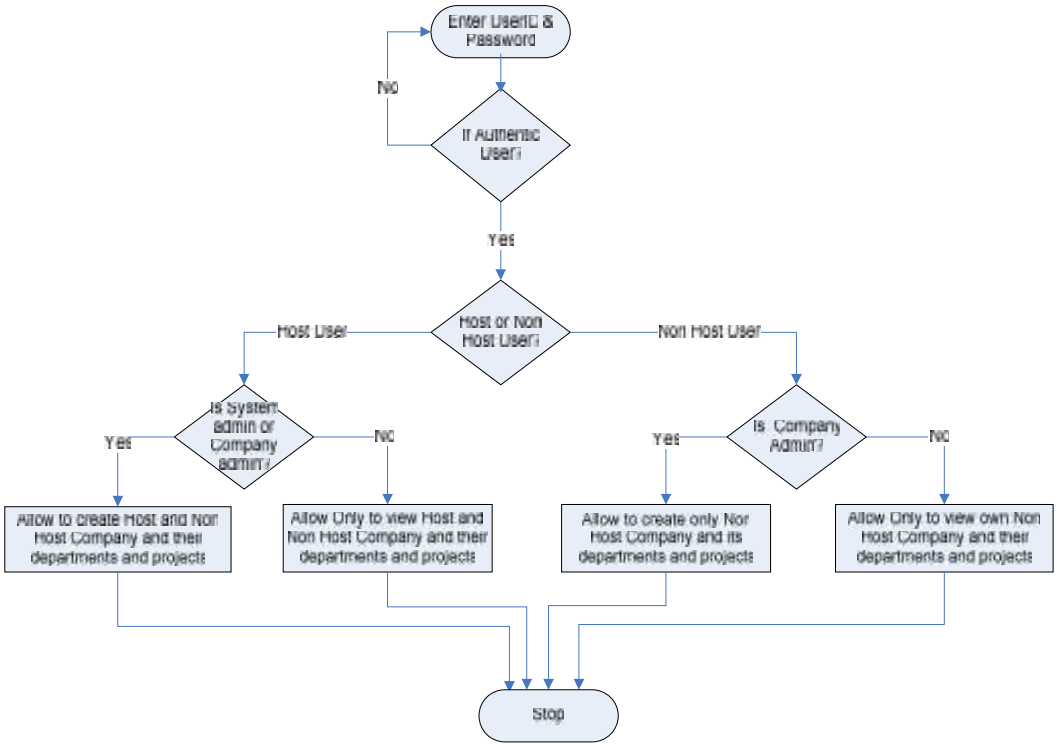

Figure 4.3.1 Access control for creating company, department and project

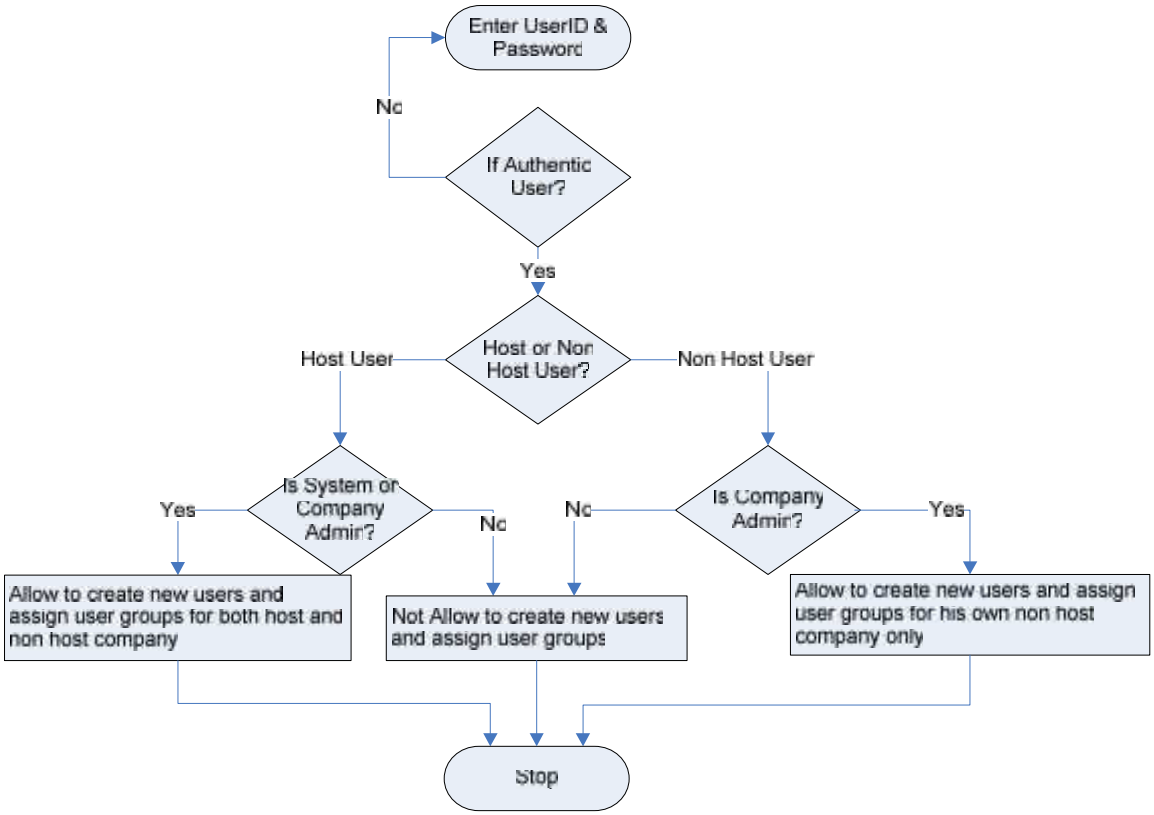

Figure 4.3.2:- Access Control for creating user and assign groups

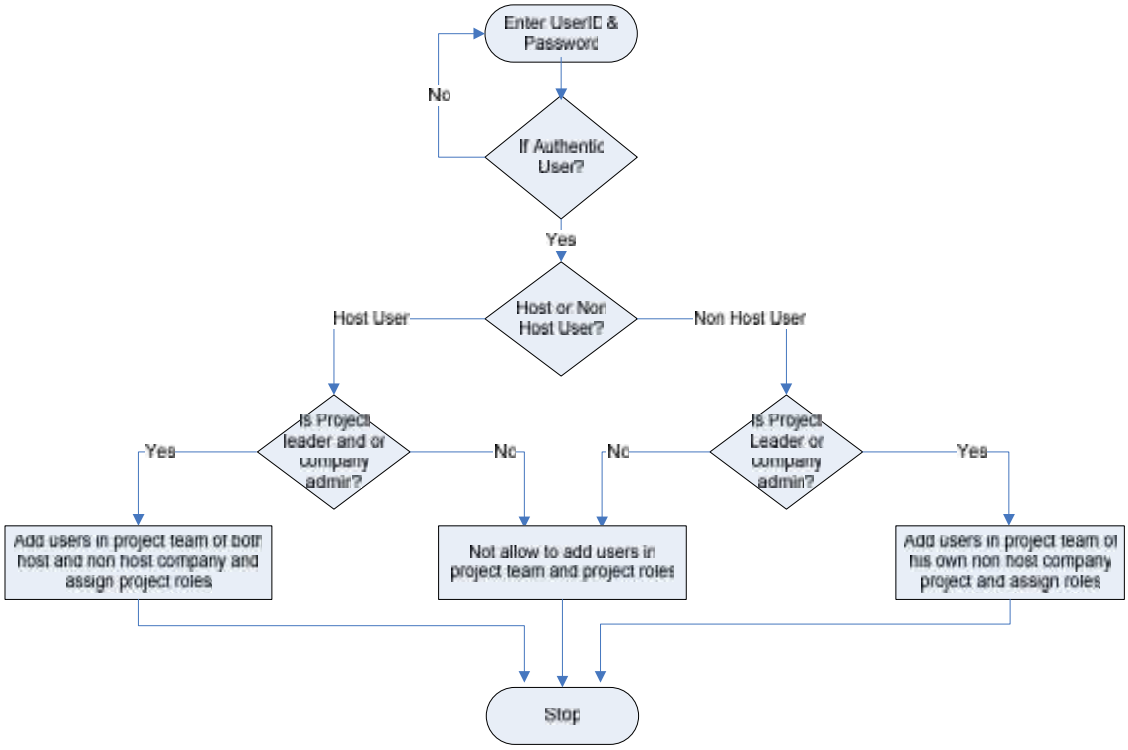

Figure 4.3.4:- Access control for adding users and assign Project roles to team-members.

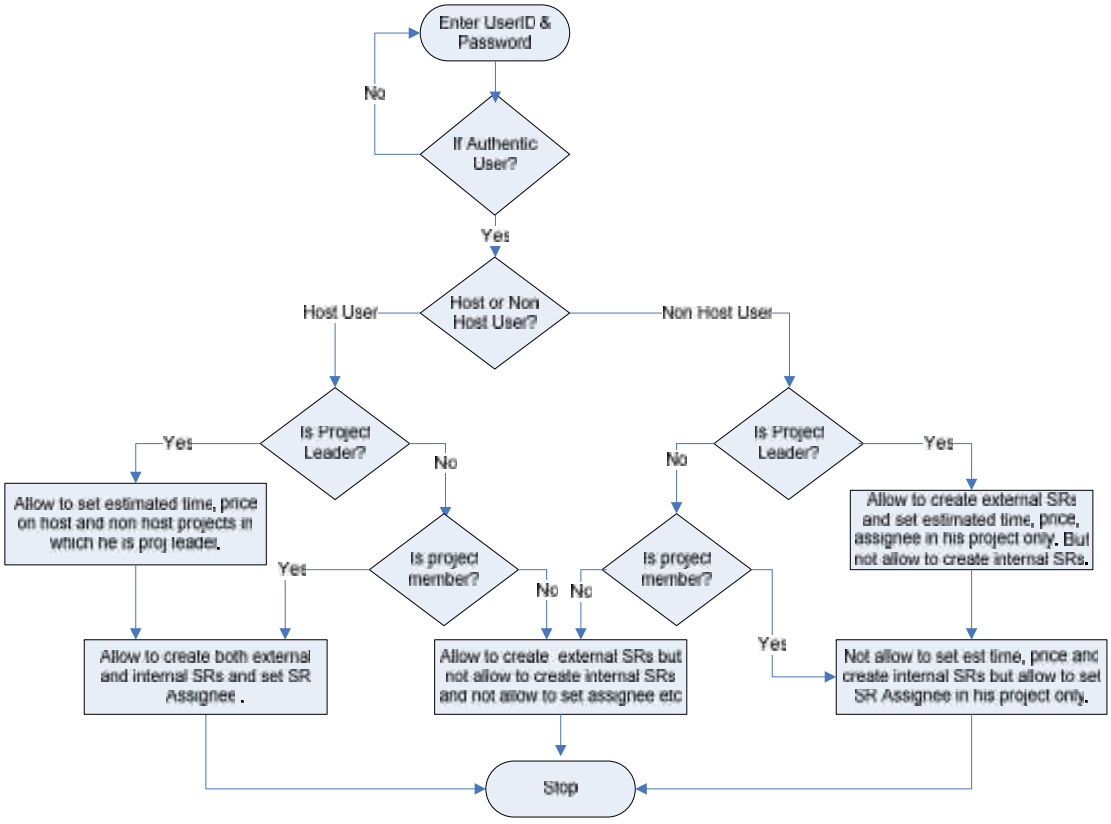

Figure 4.3.5:- Access control for creating SRs, setting and updating various values.

4.4 Security Defined for different Modules

Tables below illustrate the function access for specific Application/Project Roles. These functions will get mapped with Roles to define a Role Permission. However, the permission might have assigned to any role may limit to the either company level or project level as per assigned role permission and security issues described.

Security define for Company Module

Functions	Roles				
	System Admin	Company Admin	Project Lead	Project Mem	User
Access Company	☑	☑	☑	☑	☑
Add Companies	☑				
View own Company	☑	☑	☑	☑	☑
View other company	☑(Host)	☑(Host)	☑(Host)	☑(Host)	☑(Host)
Create host company	☑(Host)				
Create non host company	☑	☑			
Delete Companies	☑	☑			

Table 4.4.1:- Access control for company module

Security Define For Department Module

Functions	Roles				
	System Admin	Company Admin	Project Lead	Project Mem	User
Access Department Menu	☑	☑			
Create Departments	☑	☑			
Delete Departments	☑	☑			

Table 4.4.2:- Access control for department module

Security Define For User Module

Functions	Roles				
	System Admin	Company Admin	Proj Lead	Proj Mem	User
Access User Menu	☑	☑	☑	☑	☑
Create Users	☑	☑			
Update Users	☑	☑			
Delete Users	☑	☑			

Table 4.4.3:- Access control for User module

Security Define For User Group Module

Functions	Roles				
	System Admin	Company Admin	Proj Lead	Proj Mem	User
Access User Group Menu	☑	☑	☑	☑	☑
Assign User Groups	☑	☑			
Delete User Groups	☑	☑			

Table 4.4.4:- Access control for User Group module

Security Define for Project Module:

Functions	Roles				
	System Admin	Company Admin	Project Lead	Project Mem	User
Access Project Menu	☑	☑	☑	☑	☑
Create Projects	☑	☑			
Update Projects	☑	☑	☑		
Delete Projects	☑	☑			

Table 4.4.5:- Access control for Project module

Security Define for Project Team Module

Functions	Roles				
	System Admin	Company Admin	Project Lead	Project Mem	User

Access Project Team	☑	☑	☑	☑	☑
Add User to Project Team		☑	☑		
Remove User from Team		☑	☑		
Grant/Revoke Project Role to user		☑	☑		

Table 4.4.6:- Access control for Project team module

Security Define for Service Request Module

Functions	Roles				
	System Admin	Company Admin	Project Lead	Project Mem.	User
Access SR Menu	☑	☑	☑	☑	☑
Can create	☑	☑	☑	☑	☑
Delete SR			☑		
Update SR	☑	☑	☑	☑	☑
Assign Assignee, estimate etc			☑	☑	

Table 4.4.7:- Access control for Service Request module

4.5 User Interfaces

4.5.1 Company Interfaces

The following interfaces are the company interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 5.4.1. First figure is the interface for the host user where all host and non host companies are visible and if the logged in user is system or company admin then Create, Update and Delete buttons are enabled otherwise disabled. Similarly, second figure is the interface for the non host user

where only the non host company is visible and if the logged in user is company admin then only the Create, Update and Delete buttons are enabled otherwise disabled.

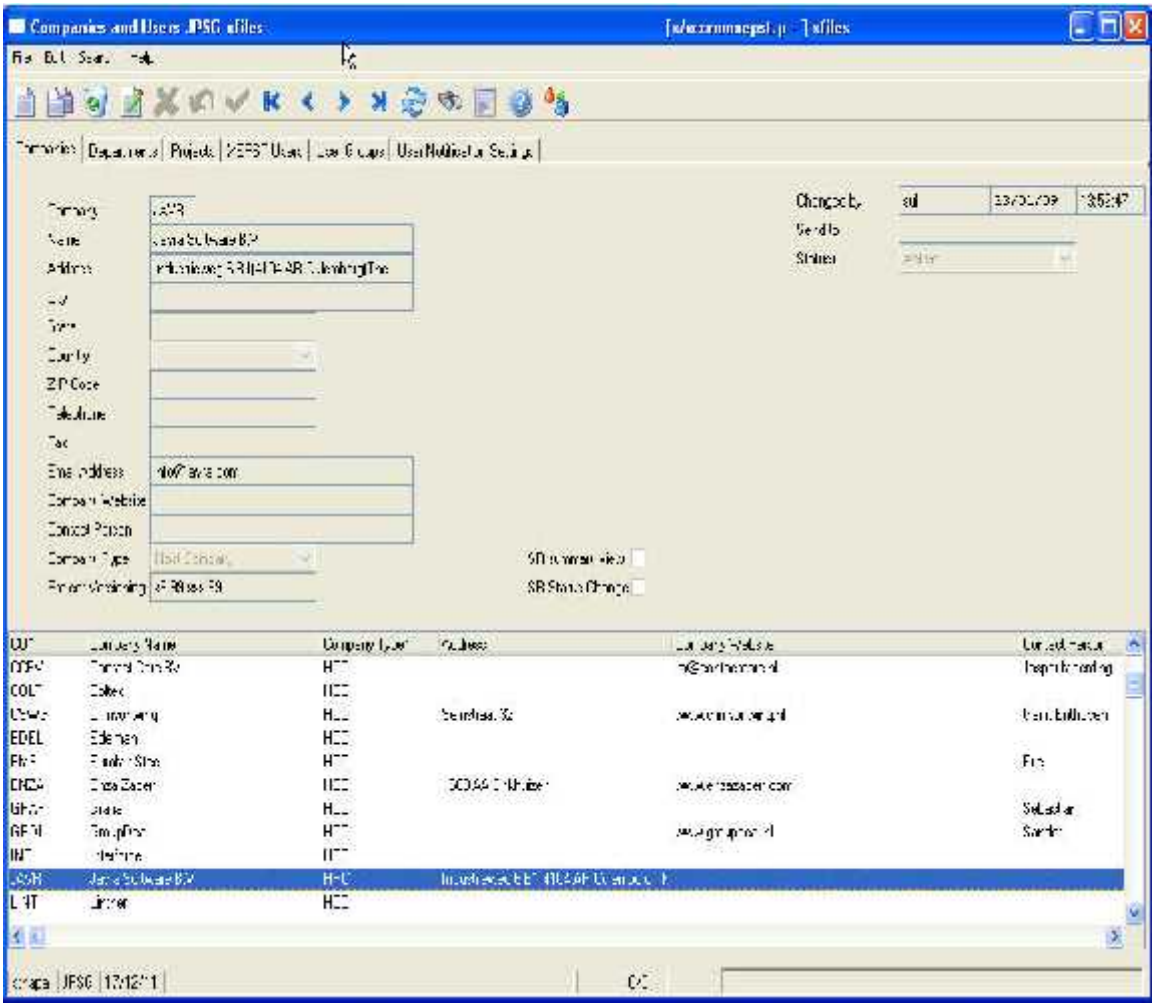

Figure 4.5.1.1:- Company Interface for Host User.

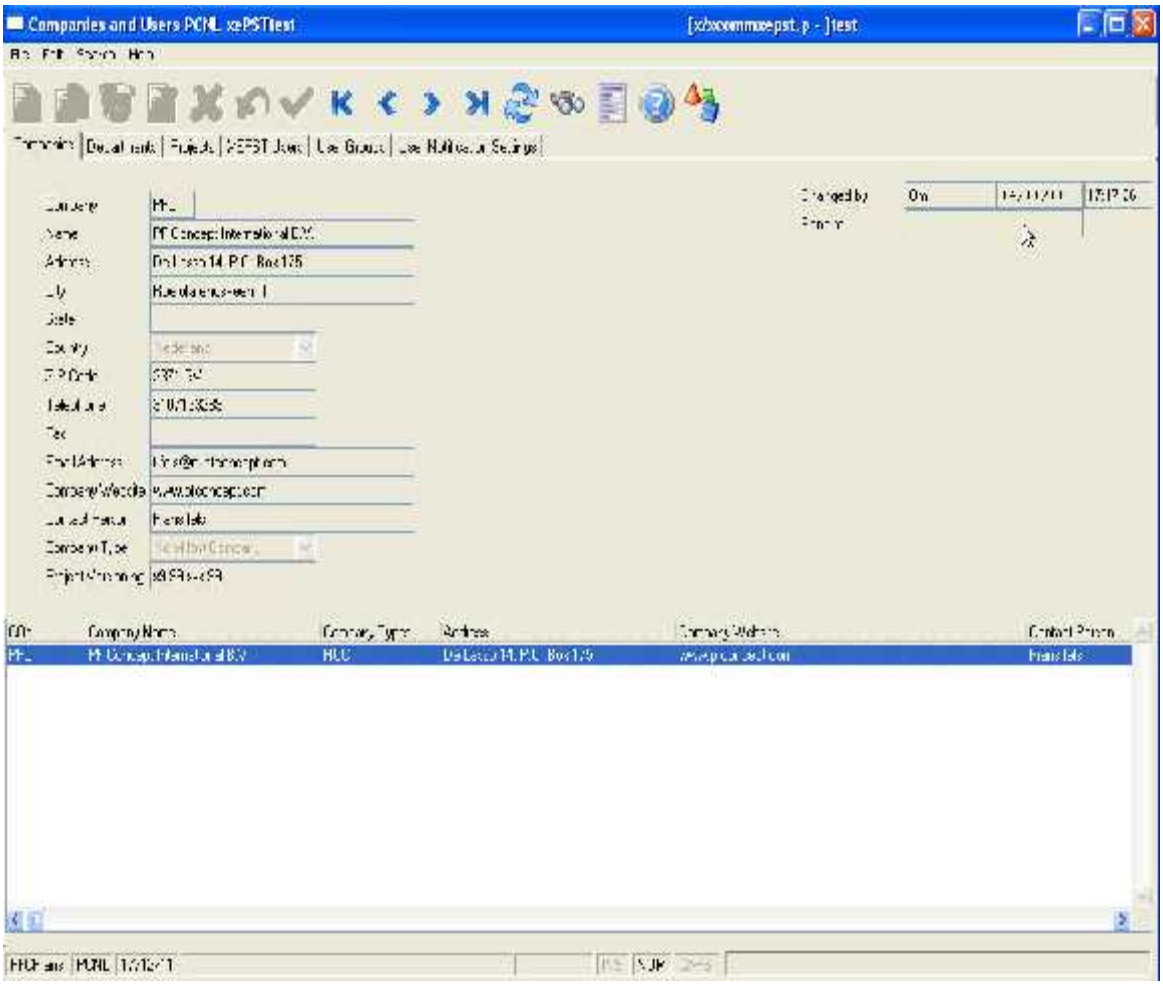

Figure 4.5.1.2:- Company Interface for Non Host User.

4.5.2 Department Interfaces

The following interfaces are the department interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 4.4.2. First figure is the interface for the host user where all departments from host and non host company are visible and if the logged in user is system or company admin then Create, Update and Delete buttons are enabled otherwise disabled. Similarly, second figure is the interface for the non host user where only the departments from non host company are visible and if the logged in user is company admin then only the Create, Update and Delete buttons are enabled otherwise disabled.

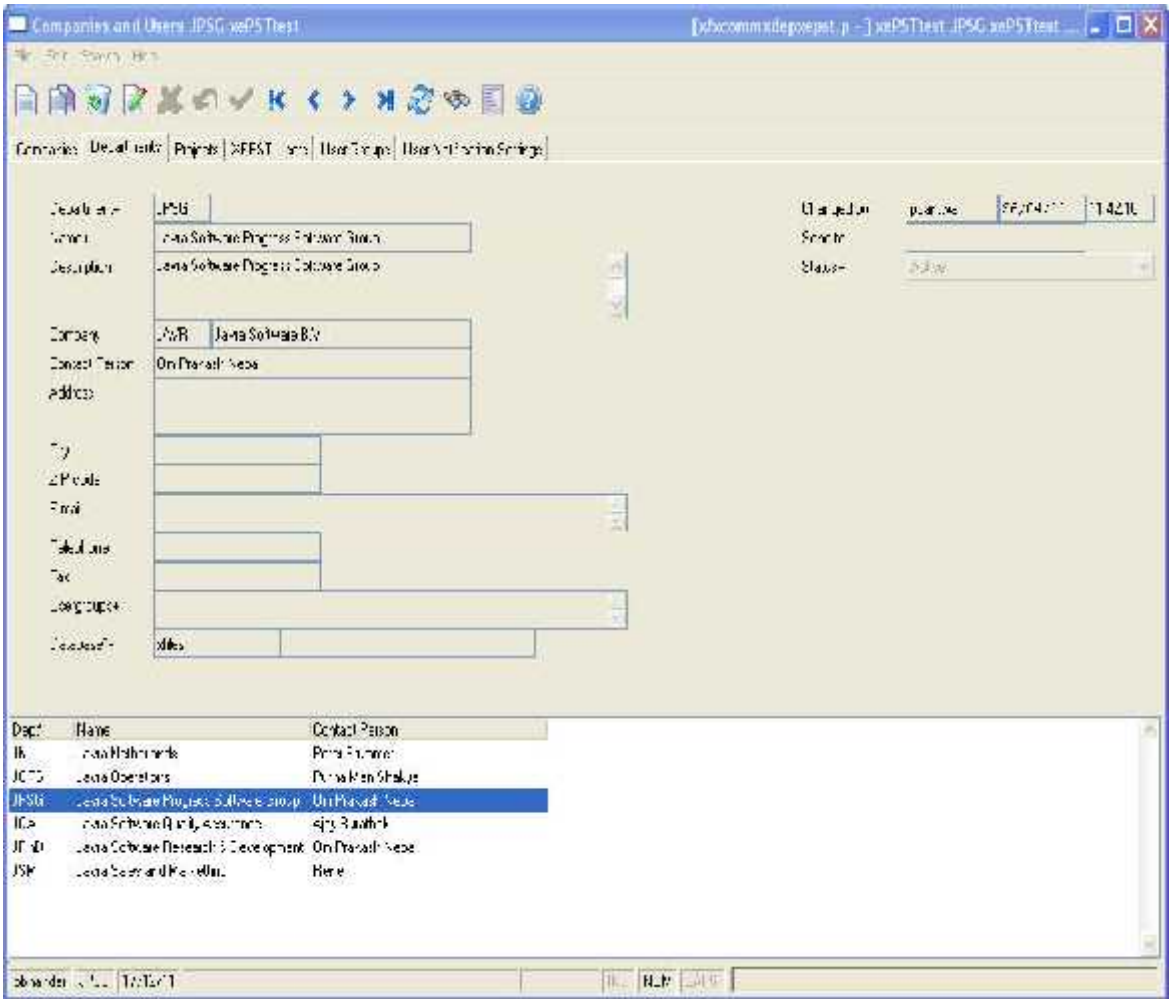

Figure 4.5.2.1:- Department Interface for Host User

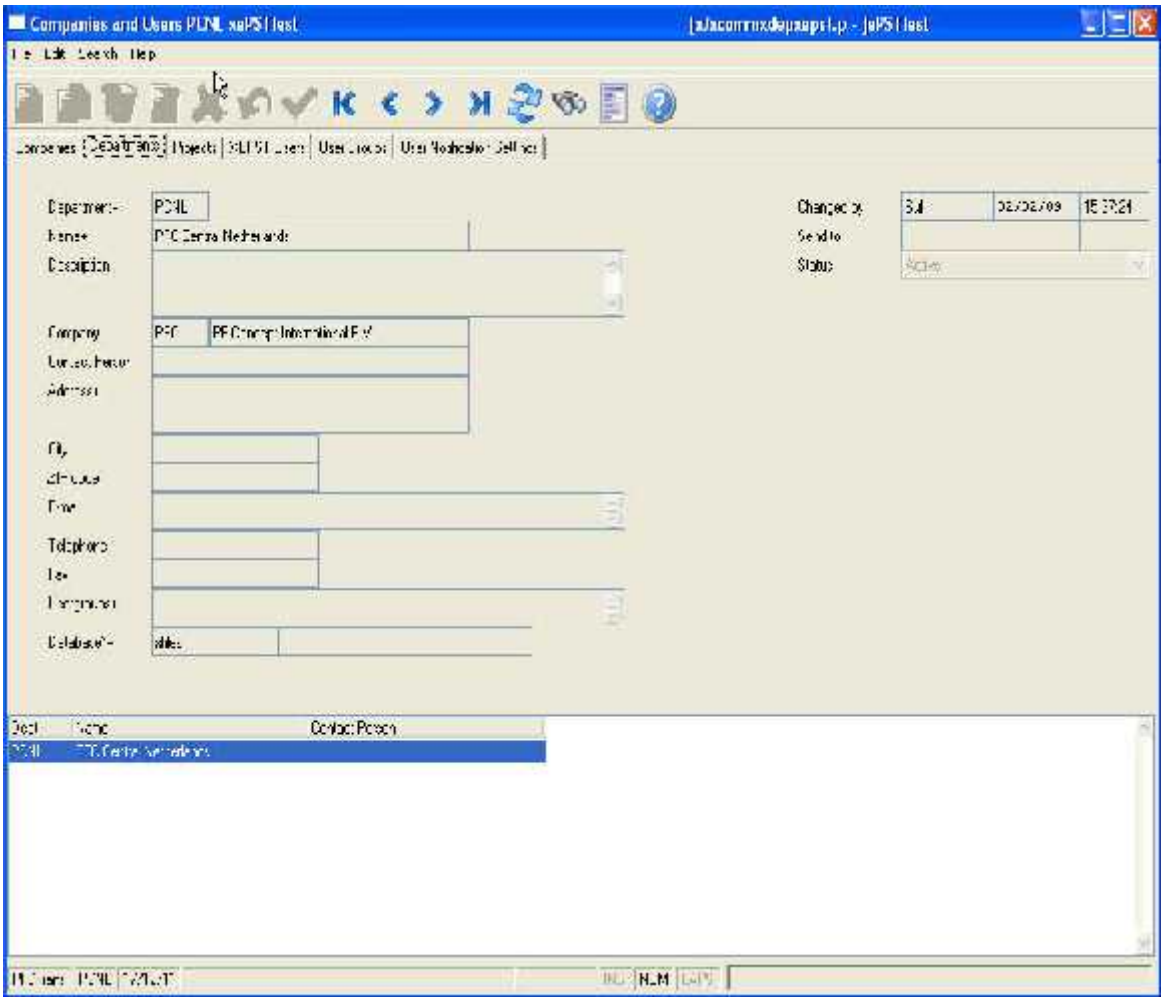

Figure 4.5.2.2:- Department Interface for Non Host User

4.5.3 Project Interfaces

The following interfaces are the project interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 4.4.3. First figure is the interface for the host user where all projects from host and non host company are visible. Similarly, second figure is the interface for the non host company user where only the projects from non host company are visible. Create, Update and Delete buttons are enabled and disabled as per the access control defined in table 4.4.3.

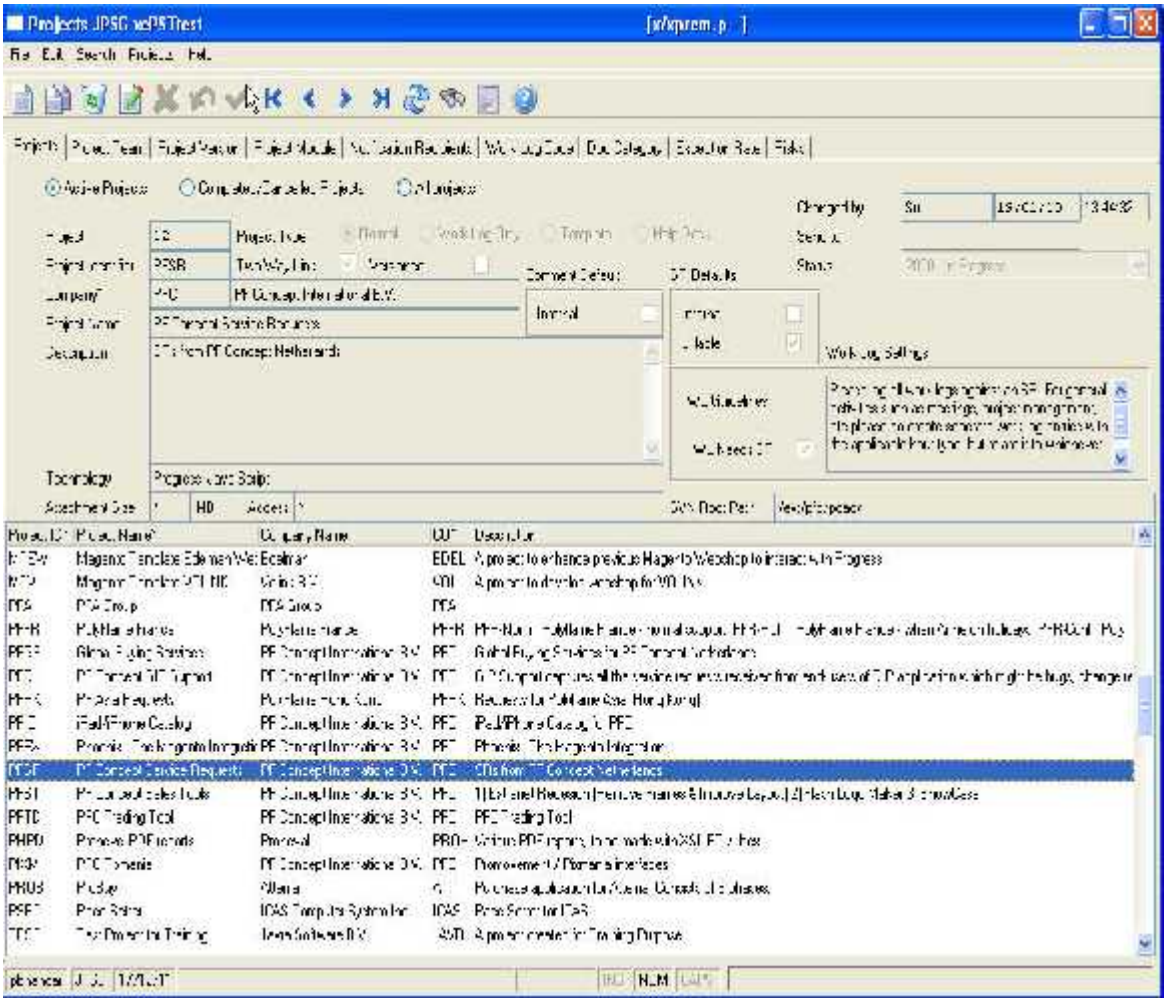

Figure 4.5.3.1:- Project Interface for Host User

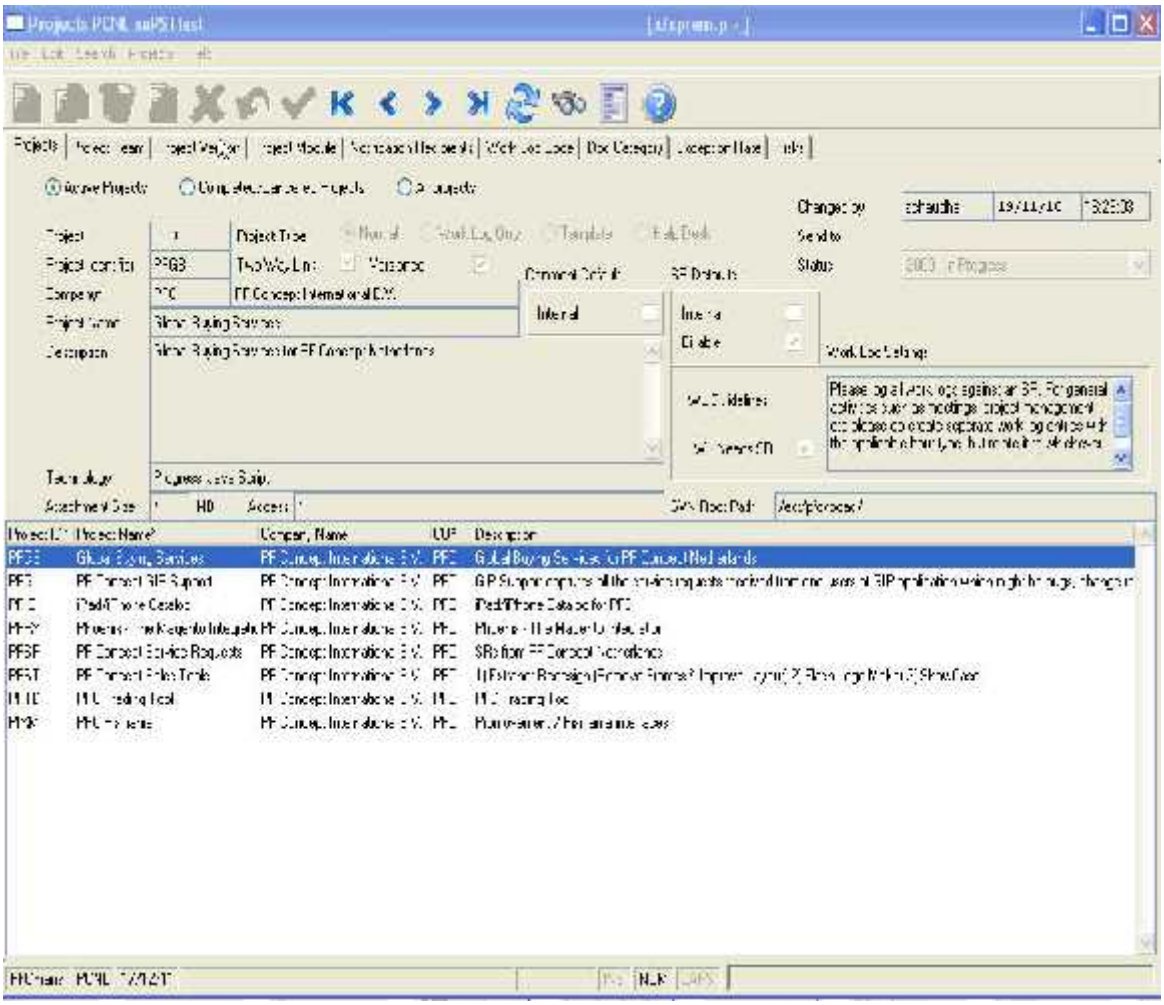

Figure 4.5.3.2:- Project Interface for Non Host User

4.5.4 User Interfaces

The following interfaces are the User interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 4.4.4. First figure is the interface for the host user where all users from host and non host company are visible. Similarly, second figure is the interface for the non host user where only users from non host company are visible. Create, Update and Delete buttons are enabled and disabled as per the access control defined in table 4.4.4.

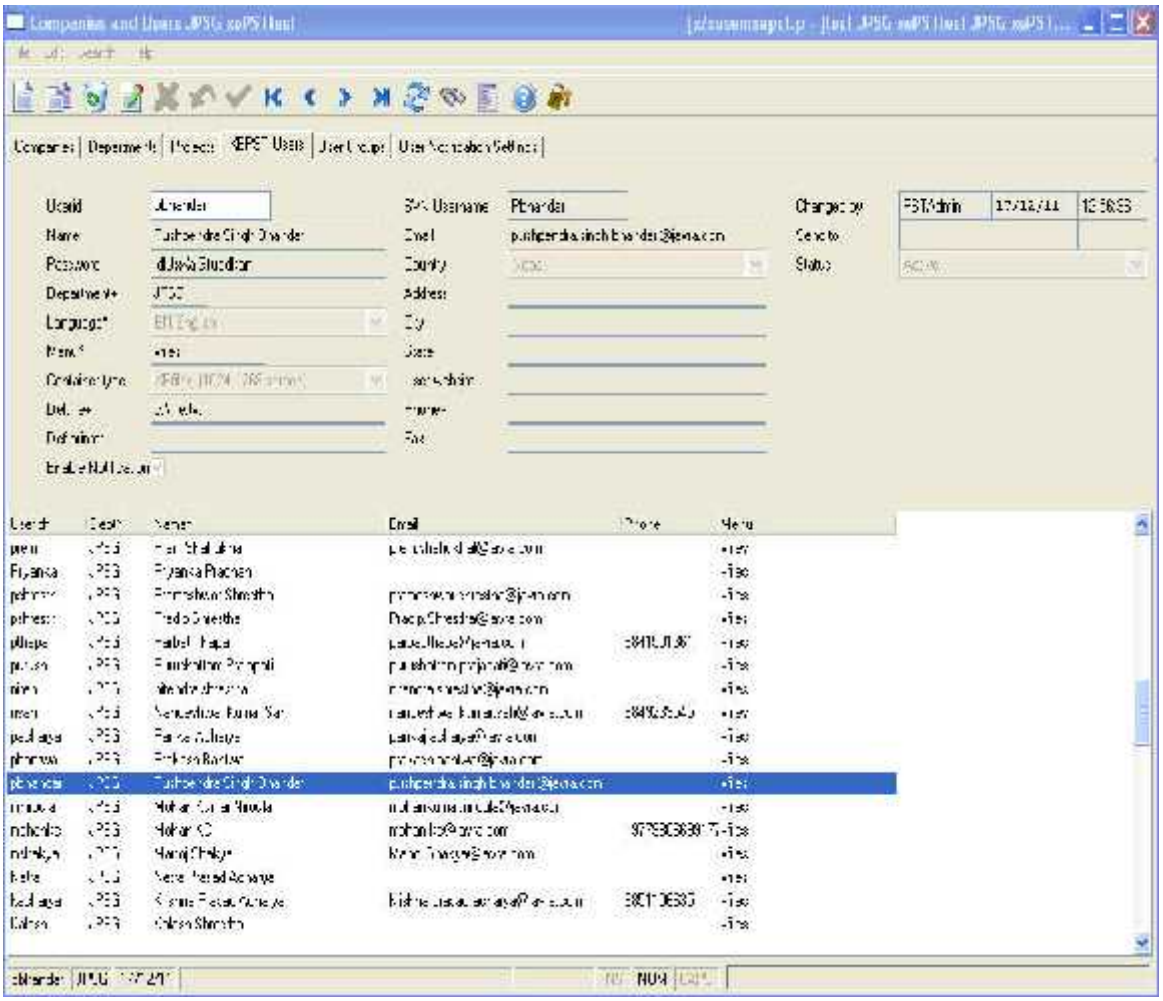

Figure 4.5.4.1:- User Interface for Host User

Figure 4.5.4.2:- User Interface for Non Host User

4.5.5 User Group Interfaces

The following interfaces are the User Group interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 4.4.5. First figure is the interface for the host user where all users from host and non host company and assigned user groups are visible. Similarly, second figure is the interface for the non host user where only users from non host company and assigned user groups are visible. Create and Delete buttons are enabled and disabled as per the access control defined in table 4.4.5.

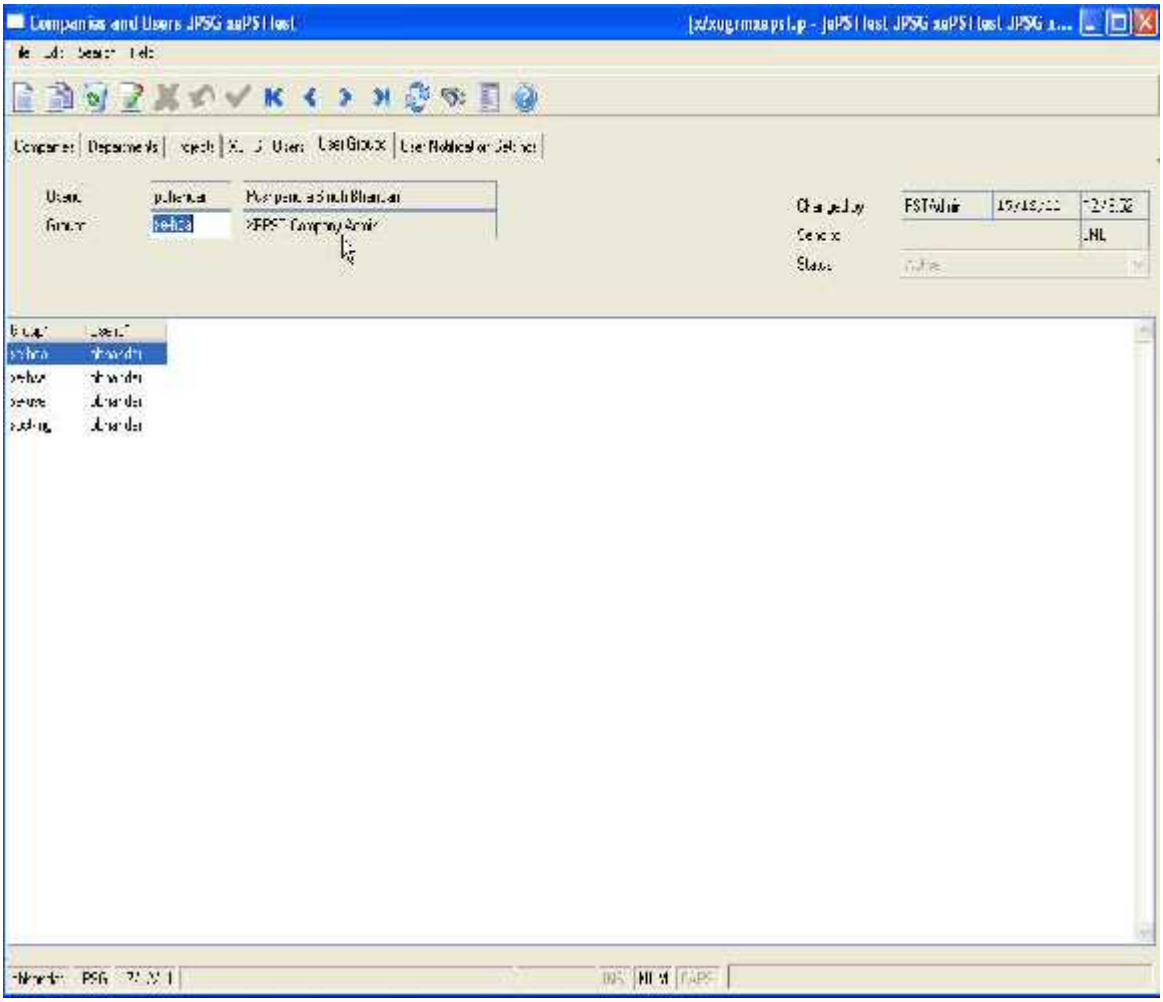

Figure 4.5.5.1:- User Group Interface for Host User

Figure 4.5.5.2:- User Group Interface for Non Host User

4.5.6 Project Team Interfaces

The following interfaces are the Project Team interfaces respectively for host (Internal) and non host user (External) based on the security defined in the table 4.4.6. First figure is the interface for host user where all the users per project of the company and different project roles are accessible. Similarly, second figure is the interface for non host user where users per project of the non host company are accessible with different project roles. Create, Update and Delete buttons are enabled and disabled as per the access control defined in table 4.4.6.

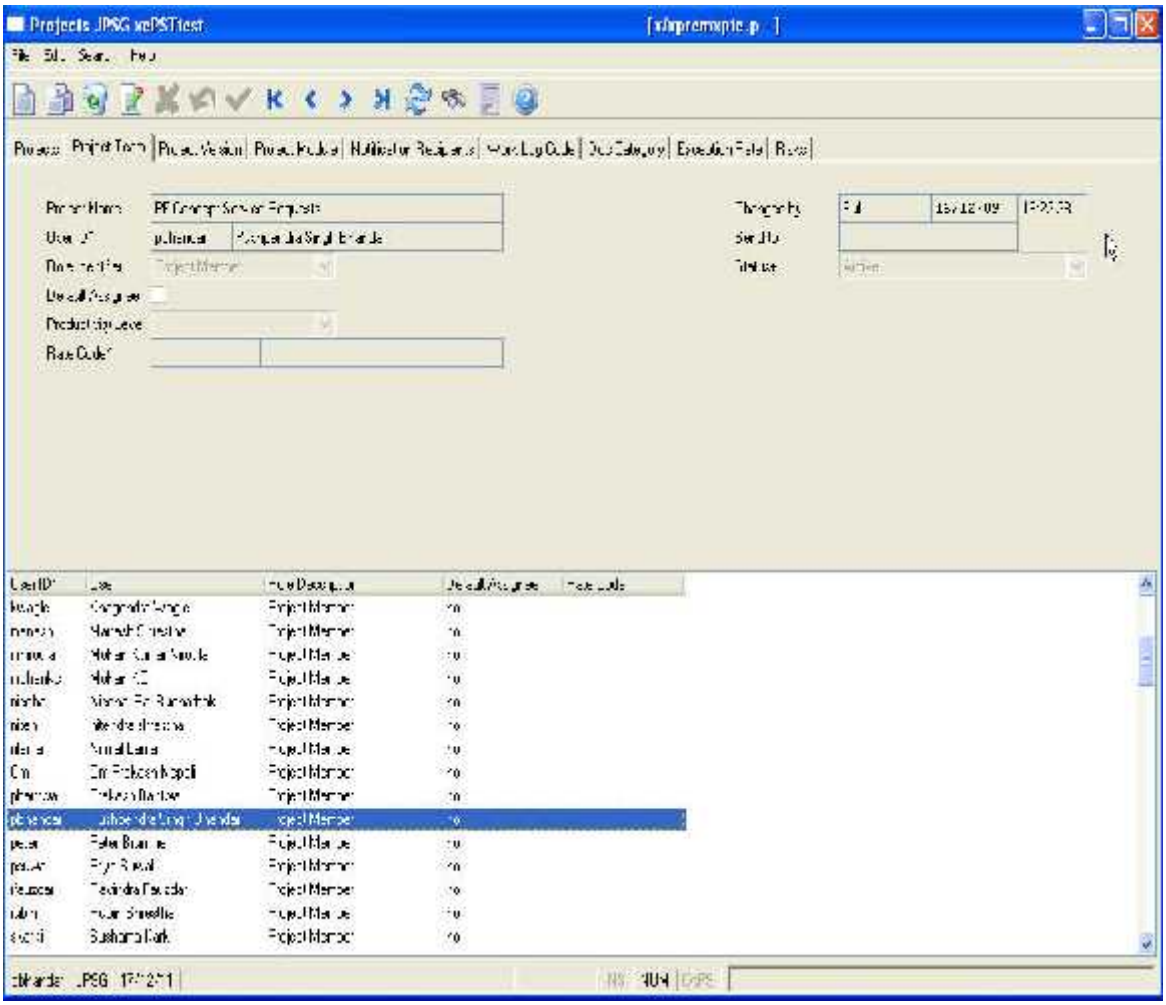

Figure 4.5.6.1:- Project Team Interface for Host User

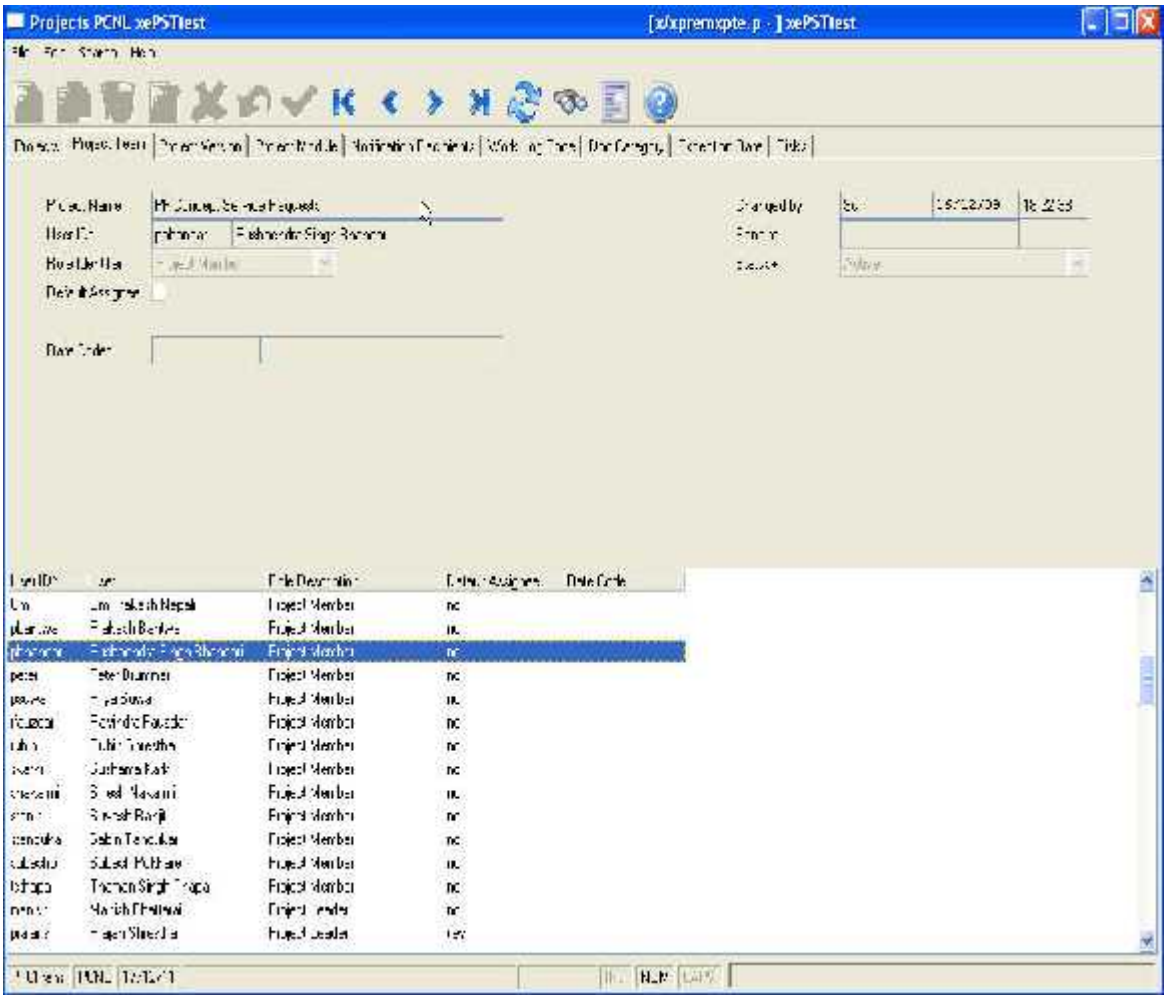

Figure 4.5.6.2:- Project Team Interface for Non Host User

4.5.7 Service Request Interfaces

The following interfaces are the Service Request interfaces respectively for both host and non host user , host (Internal) and non host user (External) based on the security defined in the table 4.4.7. First figure is the interface seen by both host and non host user in view mode. Host users can view all the Service Request from host and non host company projects whereas non host users can only view the Service Requests from non host company projects. Similarly, some of the contents which are not updatable to non host users are updatable to the host users. Second figure is the interface opened in Update mode for host user where almost every information are updatable except few which are updatable based on different role permissions. Finally, third figure is the interface opened

in update mode for non host users where project estimations and project schedule related information are not updatable. Create, Update and Delete buttons are enabled and disabled as per the access control defined in table 4.4.7.

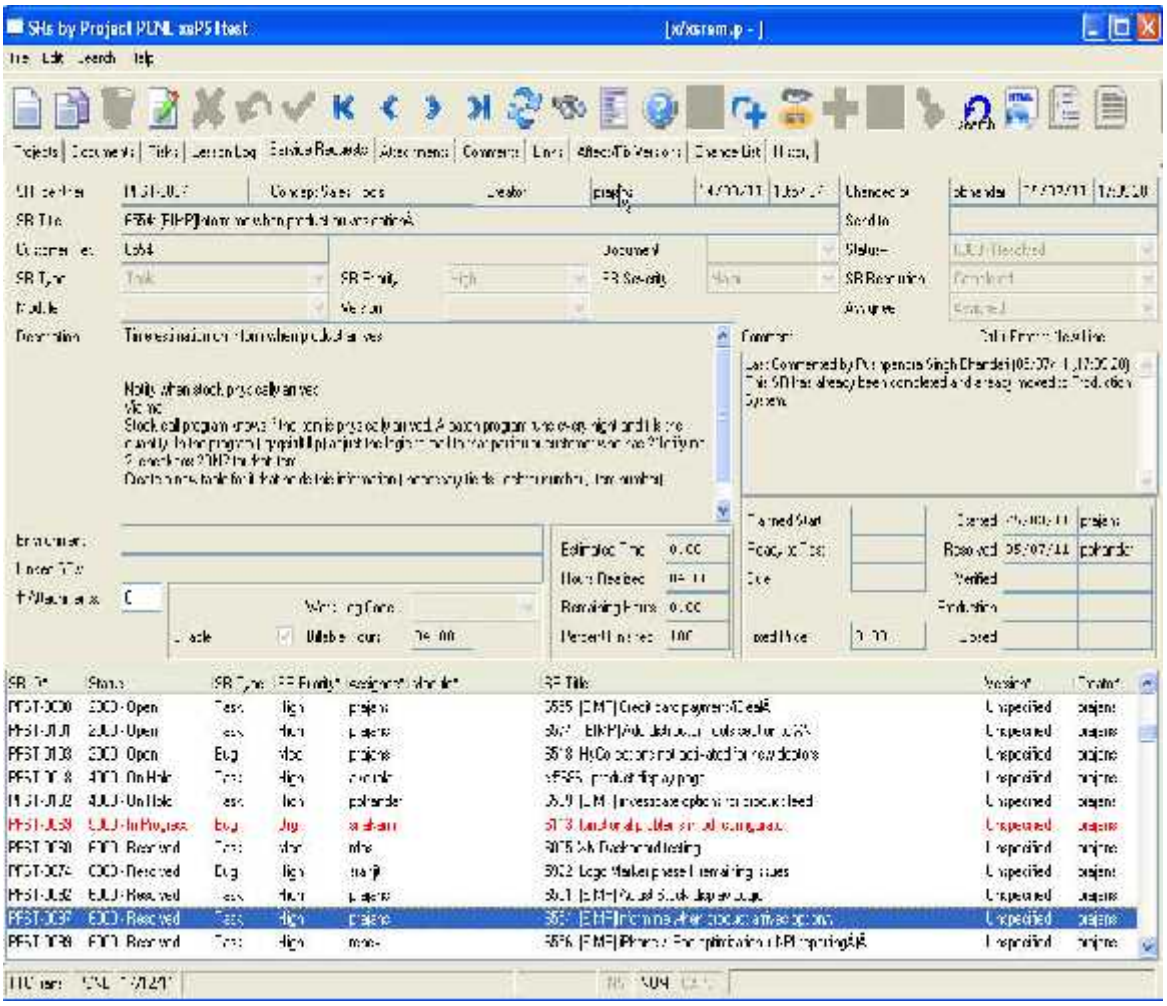

Figure 4.5.7.1:- Service Request Interface for Host and Non Host User

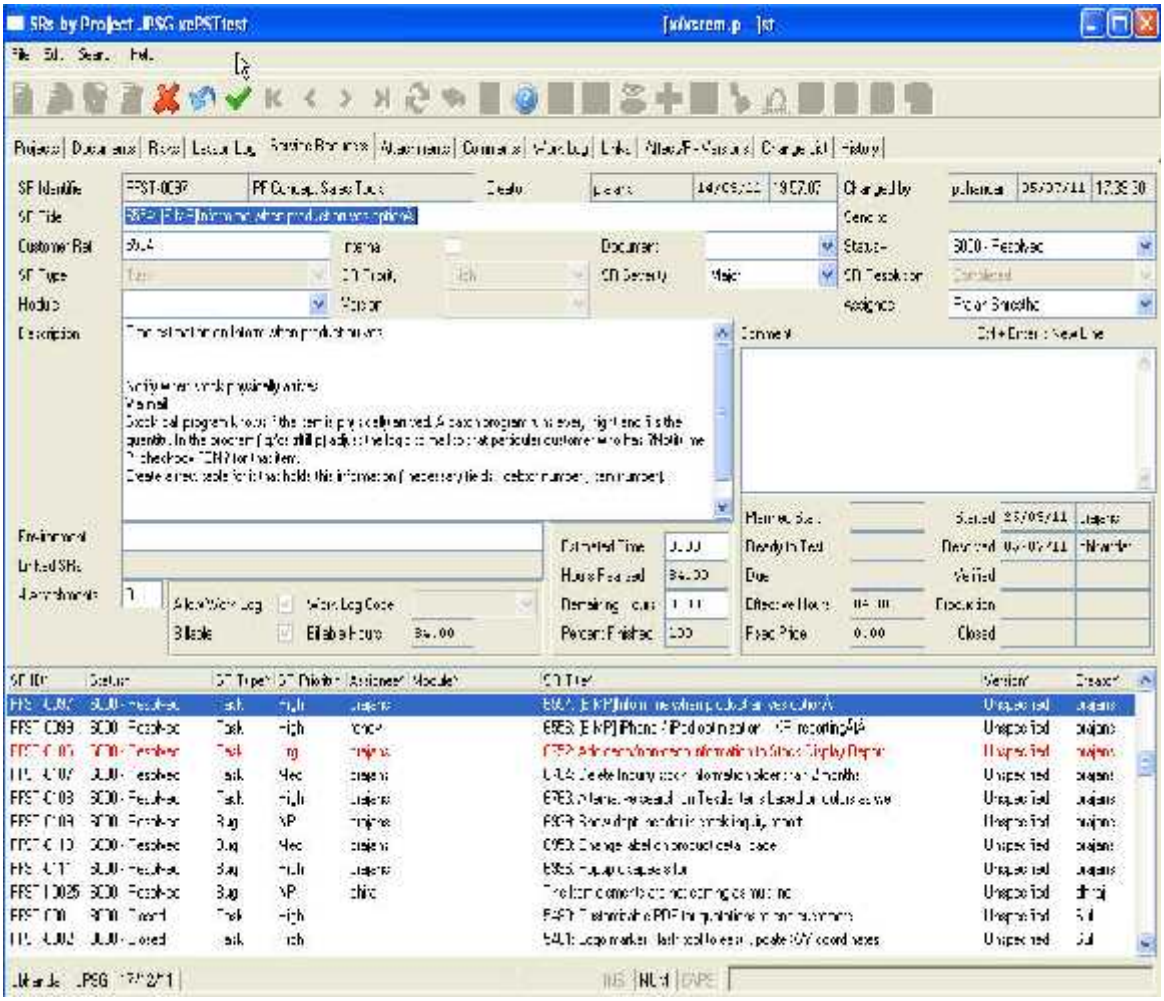

Figure 4.5.7.1:- Service Request Interface in Update Mode for Host User.

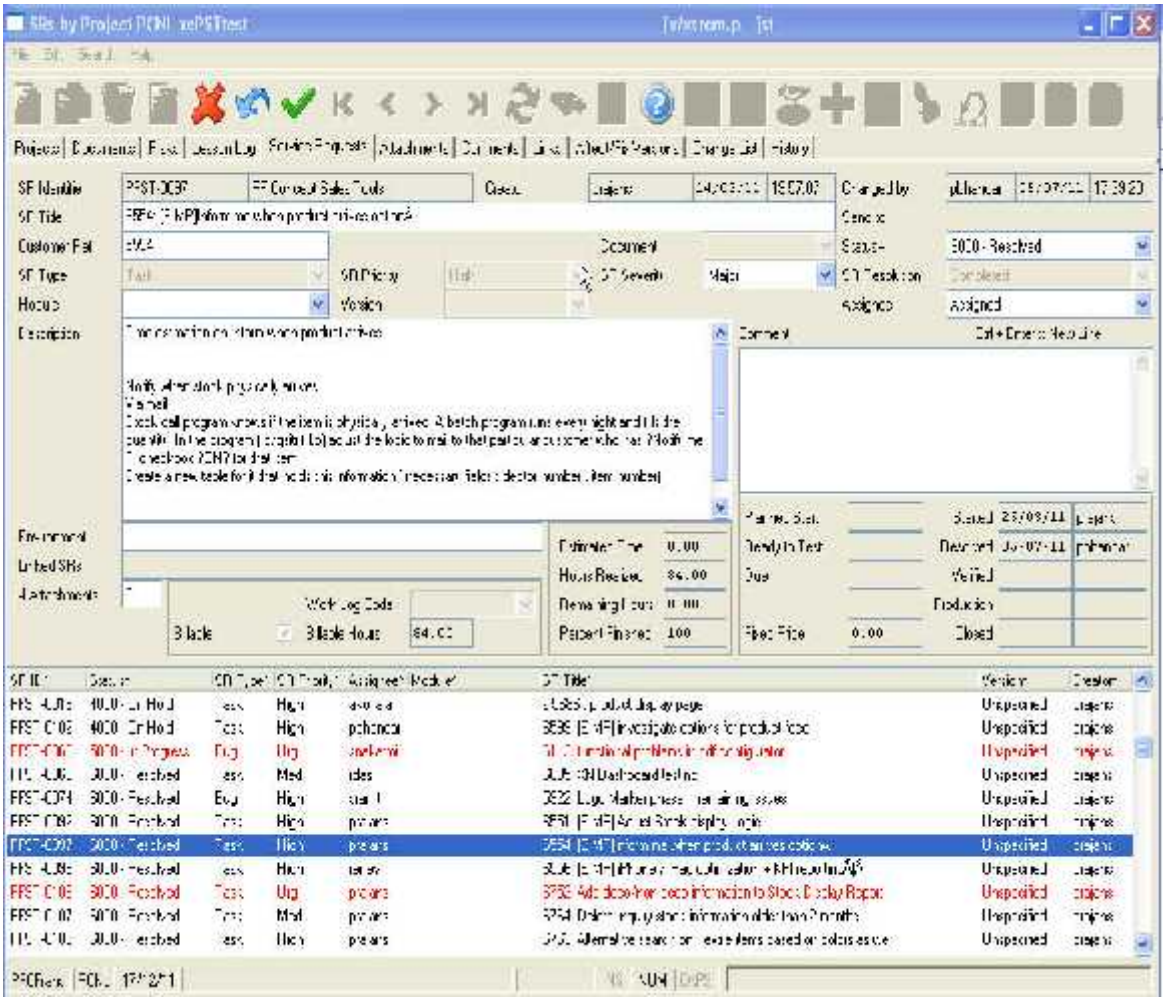

Figure 4.5.7.2:- Service Request Interface in Update Mode for Non Host User

CHAPTER 5

CONCLUSION

Authorization is a rich and complex topic, encompassing all the facets required to turn authentication and authorization into an infrastructural platform for security services. A successful deployment of authorization can, to a large extent, solve the security problems faced by many organizations today. Finally, based on the detailed analysis performed on authorization, following points have been concluded.

- The need for an authorization system in any system is known.
- Requirements for an authorization system are known and they differ based on organizations and companies.
- Different authorizations policies and techniques available are known.
- Concept of use of authorization is developed by implementing a real world application.

REFERENCES

1. Elhadi Shakshuki, Zhonghai Luo, Jing Gong 2005. An agent-based approach to security service. Journal of Network and Computer Applications 28 (2005) 183-208.
2. Singh, S. And S. Bawa 2007. A Privacy, Trust and Policy based Authorization Framework for Services in Distributed Environments. International Journal of Computer Science Vol.2 No. 2.
3. Laccetti, G. and G. Schmid. 2007. *A framework model for grid security*. Future Generation Computer Systems 23, 702–713.
4. R. Alfieri, R. Cecchini, V. Ciaschini, L.dell' Agnello,A.Frohner, K.Lorenty, F. Spataro 2005, 'From Gridmap-File to VOMS: managing authorization in Grid Environment'. Future Generation Computer Systems 21 (2005) 549-558.
5. Ashley, P., M. Vandenwauver and F. Siebenlist. 2000. Applying authorization to intranets: architectures, issues and APIs. Computer Communications 23 (2000) 1613–1620.
6. Leach P, Kaler C, Dillaway Blair, Garg P, LaMacchina B, Lampson B, Manferdelli J, Rashis R, Shewchuk J, Simon D, Ward R. A Conceptual Authorization for Web Services.
7. Scavo, T., S. Cantor. Shibboleth Architecture Technical Overview. http://shibboleth.internet2.edu/shibbolethdocuments.html
8. Welch, Von, Tom Barton, Kate Keahey, Frank Siebenlist. "Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Collaboration." Proceedings of the 4th Annual PKI R&D Workshop, 2005.
9. N. Papatheodoulou, N. Skalvos,IEEE Member, 2009, Architecture & System Design of Authentication, Authorization & Accounting Services: Proceedings of The IEEE Region 8, EUROCON 2009, International Conference (IEEE EUROCON'09),St. Petersburg, Russia, May 18-23, 2009.
10. 2008 IEEE Congress on Services –Part I, Martino A. S. A Model for Securing E-Banking Authentication Process: Antiphishing Approach.

11. Housley R, Polk W, Ford W, Solo D. 2002. *Internet public key infrastructure, Part I*: X.509 certificate and CRL profile. Request for Comments (RFC) 3280. In: Journal of Network and Computer Applications 30 (2007) 900–919.
12. Lambrinoudakis C, Gritzalis S, Dridi F, Pernul G. Security requirements for e-government services: a methodological approach for developing a common PKI-based security policy.
13. Understanding PKI, 2002: Concepts, Standards, and Deployment Considerations-Adams C, Lloyd S, Addission Wesley.
14. Saxena, A. 2004. Public key Infrastructure: concepts, Design and Deployment-. Tata McGraw Hill.
15. Andrew S. Tanenbaum, "Computer Networks", Third Edition, PrenticeHall of India.
16. Behrouz A. Forouzan, "Data Communications and Networking", Fourth Edition, Tata McGraw-Hill Publishing.
17. N.Li, D.Lee, "Virtual Authentication Ring for Securing Network Operations ", Proceedings of NTMS 2007 Conference, 2007.
18. N.Sklavos. X.Zhang. "Wireless Security & Crytography: Specifications and Implementations", CRC-Press, A Taylor and Francis Group, 2007.
19. J. Vollbrecht, "AAA Authorization Framework", RFC 2904,IETF, August,2000.
20. http://en.wikipedia.org/wiki/Authorization
21. http://en.wikipedia.org/wiki/Identity_management
22. http://en.wikipedia.org/wiki/Access_control
23. http://en.wikipedia.org/wiki/Shibboleth_%28Internet2%29