Database Security & Access Control Models: A Brief Overview

Kriti, Indu Kashyap
CSE Dept.
Manav Rachna International University, Faridabad, India

Abstract

Database security is a growing concern evidenced by increase in number of reported incidents of loss of or unauthorized exposure of sensitive data. Security models are the basic theoretical tool to start with when developing a security system. These models enforce security policies which are governing rules adopted by any organization. Access control models are security models whose purpose is to limit the activities of legitimate users. The main types of access control include discretionary, mandatory and role based. All the three techniques have their drawbacks and benefits. The selection of a proper access control model depends on the requirement and the type of attacks to which the system is vulnerable. The aim of this paper is to give brief information on database security threats and discusses the three models of access control DAC, MAC & RBAC.

1. Introduction

Information is a critical resource in today's enterprise, whether it is industrial, commercial, educational etc. As the organizations increase their adoption of database systems as the key data management technology for day-to-day operations and decision making, the security of data becomes crucial [14]. The Defence Information System Agency of US Department of defence states that database security should provide controlled, protected access to the contents of database as well as preserve the integrity, consistency and overall quality of the data [2].

Database security encompasses three constructs [1]: confidentiality, integrity, and availability.

- **Confidentiality:** Protection of data from unauthorized disclosure.

- **Integrity:** Prevention from unauthorized data access.

- Availability: Identification and recovery from hardware and software errors or malicious activity resulting in the denial of data availability.

1.1 Threats to Database Security

A threat can be identified with a hostile agent who either accidently or intentionally gains an unauthorized access to the protected database resource [3] [4].

In organizations there are top ten type of threats are recognized with can't only increase the risk of database exposure but also cause disastrous consequences on the entire organization. Some of these threats are described below [19]:

Threat 1 - Excessive Privilege Abuse

When users (or applications) are granted database access privileges that exceed the requirements of their job function, these privileges may be abused for malicious purpose.

Threat 2 - Legitimate Privilege Abuse

Users may also abuse legitimate database privileges for unauthorized purposes.

Threat 3 - Platform Vulnerabilities

Vulnerabilities in underlying operating systems may lead to unauthorized access, data corruption, or denial of service.

Threat 4 - SQL Injection

In a SQL injection attack, attacker typically inserts unauthorized database statements into a vulnerable SQL data channel.

Threat 5 - Denial of Service

Denial of Service (DOS) is a general attack category in which access data is denied to intended users.

Threat 6 - Weak Authentication

Weak authentication schemes allow attackers to assume the identity of legitimate database users by stealing or otherwise obtaining login credentials.

1.2 Database Security Policies

To eliminate threats, it is necessary to define proper security policy. Security policies are governing principles adopted by organizations [3]. They capture the security requirements of an organization, specify what security properties the system must provide and describe steps an organization must take to achieve security.

The following list gives features of a security policy for databases:

- **Access Control Policy:** These policies ensure that direct access to the system objects should proceed according to the privileges and the access rules.
- **Inference Policy:** These policies specify how to protect classified information from disclosure when the information is released indirectly in the form of statistical data.
- **User identification/authentication policy:** This policy indicates the requirements for correct identification of users. The user identification is the basis of every security mechanism. A user is allowed to access data after identification as an authorized user only.
- **Accountability and audit policy:** This policy provides the requirements for the record keeping of all accesses to the database.
- **Consistency policy:** This policy defines the state in which the database is considered valid or correct and includes operational, semantic and physical integrity of database.

1.3 Database Security Models

Security models are the formal description of security policies. Security models are useful tools for evaluating and comparing security policies [6]. Security models allow us to test security policies for completeness and consistency. They describe what mechanisms are necessary to implement a security policy.

Security models are described in terms of the following elements:

- **Subjects:** Entities that request access to objects.
- **Objects:** Entities for which access request is being made by subjects.
- **Access Modes:** Type of operation performed by subject on object (read, write, create etc.).
- **Policies:** Enterprise wide accepted security rules.
- **Authorizations:** Specification of access modes for each subject on each object.
- **Administrative Rights:** Who has rights in system administration and what responsibilities administrators have.
- **Axioms:** Basic working assumptions.

2. Access Control

The purpose of access control is to limit the actions or operations that a legitimate user of a computer system can perform. Access control constraints what a user can do directly, as well as what programs executing on behalf of the users are allowed to do. In this way access control seeks to prevent activity that could lead to a breach of security.

There are two classes of resources in any computer system: (active) subjects and (passive) objects. The ways a subject access an object are called access privileges. Access privileges allow subjects to either manipulate objects (read, write, execute, etc.) or modify the access control information (transfer ownership, grant and revoke privileges, etc.).

The access control may be based on different policies which in turn follows different principles. The choice of a security policy is important because it influences the flexibility, usability, and performance of the system. The principles on which policies are based are as follows [3]:

- **Minimum vs. Maximum privilege principle:** According to this subjects should use the minimum set of privileges necessary for their activity. The opposite of this is maximum privilege principle which is based on the principle of maximum availability of data in a database.
- **Open vs. Closed system principle:** In an open system, all accesses that are not explicitly forbidden are allowed. While in a closed system, all accesses are allowed only if explicitly authorized. A closed system is inherently more secure.
- **Centralized vs. Decentralized administration principle:** The principle addresses the issue who is responsible for the maintenance and management of privileges in the access control model. In centralized administration, a single authority controls all security aspects of the system, while in decentralized system different authorities control different portions of the database.

Access control is enforced by a reference monitor which mediates every attempted access by a user to objects in the system. The reference monitor consults an authorization database in order to determine if the user attempting to do an operation is actually authorized to perform that operation. Authorizations in this database are administered and maintained by a security administrator as shown in fig 1. The administrator sets these authorizations on the basis of the security policy.

Access control is different from authentication. Authentication is a process of signing on to a computer system by providing an identifier and a password. So, correctly establishing the identity of the user is the responsibility of the authentication service. On the other hand, access control assumes that authentication of the user has been successfully verified prior to enforcement of access control via a reference monitor.

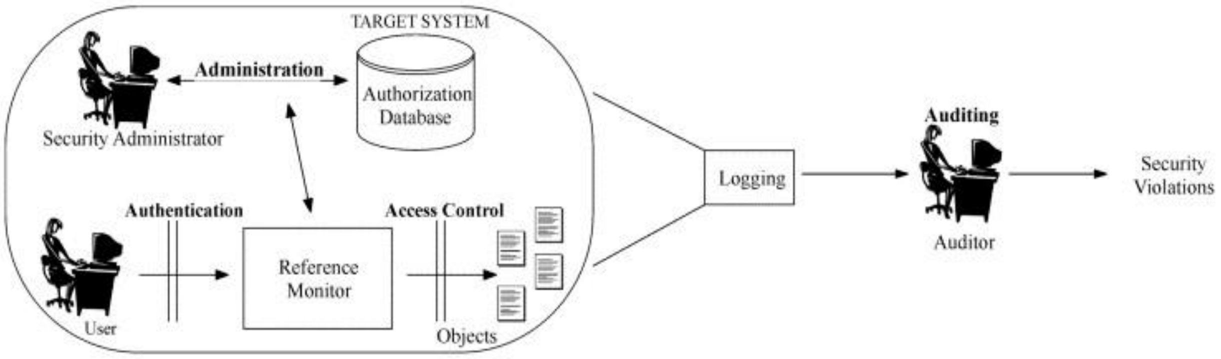

Fig 1: Access Control and Security Services

3. Discretionary Access Control (DAC)

Specify the rules, under which subjects can, at their discretion, create and delete objects, and grant and revoke authorizations for accessing objects to others [6]. That is:

Govern the access of users to information on the basis of user’s identity and predefined discretionary rules” defined by security administrator.

The rules specify, for each user and object in the system, the types of access the user is allowed for the object.

The request of a user to access an object is checked against the specified authorizations; if there exists an authorization stating that the user can access the object in the specific mode, the access is granted; otherwise it is denied. The policies are discretionary in that they allow users to grant other users authorizations to access the objects.

Discretionary policies are used in commercial systems for their flexibility, which makes them suitable for a variety of environments with different protection requirements.

Discretionary models are applied in relational database systems using System R Authorization Model and some extensions to it.

3.1 The System R Authorization Model

This model is first defined by Griffiths and Wade, and later revised by Fagin. Possible relational databases privileges user can exercise on tables are select, insert, delete and update [5].

The model supports decentralized administration of authorizations. Any database user is authorized to create a new table; once the table is created he becomes the owner of the table and is fully authorized to exercise all privileges on the table. The owner can also grant all privileges on the table to other users.

The authorization can be modeled as a tuple of the form < s, p, t, ts, g, go> stating that the user s has been granted privilege p on table t by user g at time ts. If go=”yes”, s has the grant option and, therefore, s is authorized to grant other users privilege p on table t, with or without grant option. Every time a privilege is revoked from a user, a recursive revocation may take place to delete all authorizations which would have not existed had the revokee never received any authorizations for the privilege on the table from the revoker.

To illustrate this concept, consider the consequence of grant operations for privilege p on table t illustrated in fig. 2(a), where every node represents a user and an arc between node u1 and u2 indicates that u1 granted the privilege on the table to u2. The label of the arc indicates the time the privilege was granted.

Suppose now Bob revokes the privilege on the table from David. According to the semantics of recursive revocation since David has never received authorization from Bob, David could not have granted the privilege to Ellen and Ellen to Jim. However in the diagram authorization from David to Frank does not have to be deleted because it may be granted by Chris. The set of authorizations holding in the system after the revocation is shown in fig. 2(b).

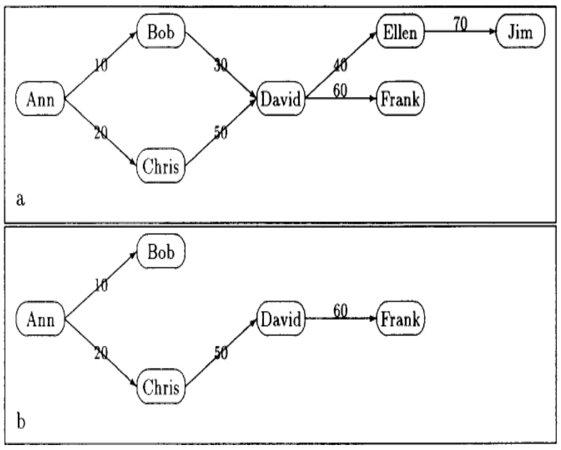

Fig 2: Bob revokes the privilege from David

3.2 Extensions to the System R Model

The previous model was extended by Wilms and Lindsay to simplify the management of authorizations [6]. They have included the authorizations for group. Group is a set of users or other groups. Group is disjoint i.e. a user or a group may belong to more than one group. Authorizations specified for group, means that they are valid for all members of the group.

The two main extensions are as follows:

- A new type of REVOKE operation, called non-cascading is introduced. Suppose now if Bob invokes the non-cascading revoke operation on the privilege granted to David, the authorization given by David to Ellen and Frank are re-specified with Bob as grantor and Jim retains the authorization given him by Ellen as shown in fig. 3(b).

- The system R model uses closed world policy under which when a user tries to access a table and if positive authorization is not found, the user is denied access. The major problem with this approach is that a user does not guarantee that he will not acquire the authorization anytime in future. The use of explicit negative authorizations can overcome this drawback. According to which if a user has both negative and positive authorization for a given privilege in the same table; the user is prevented from using the privilege on the table.

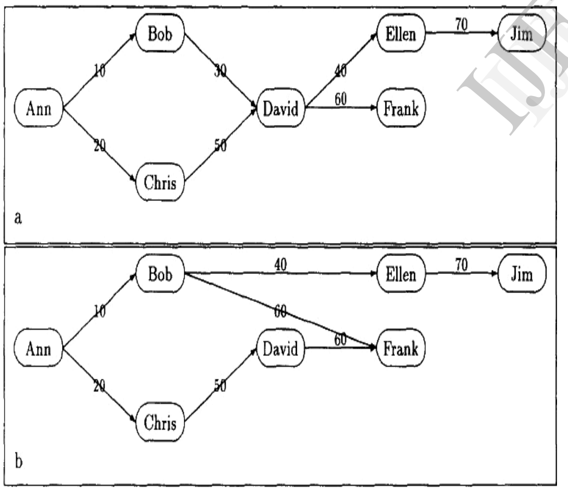

Fig 3: Bob revokes the privilege from David without cascade

3.3 Trojan Horse Attacks

The main drawback of DAC is that although each access is controlled and allowed only if authorized, it is possible to bypass the access restrictions stated through the authorizations. A subject who is able to read data can pass the data to other subjects not authorized to read the data without the cognizance of the data owner [15]. This weakness makes DAC vulnerable to malicious attacks such as Trojan Horses [6] [8].

A Trojan horse is a computer program with an apparently or actually useful function, which contains additional hidden functions that surreptitiously exploit the legitimate authorizations of the invoking process. The Trojan Horse Attacks can be understood by the example of an organization. Suppose a top level manager named Ann creates a table market containing sensitive information. Tom, one of Ann's subordinate, who also works secretly for another organisation wants this information. To achieve this Tom secretly creates a table stolen and give write privilege to Ann on this table. Ann doesn't even know about the existence of this table and having privilege on the table. Tom also secretly modifies worksheet application to include two hidden operations:

- A read operation on the table market.
- A write operation on the table stolen.

As shown in fig. 4(a). Tom then gives this new application to his manager Ann. As a result during execution sensitive information is copied from market to stolen and becomes available to dishonest employee Tom who can now misuse this information (fig. 4(b)).

4. MANDATORY ACCESS CONTROL (MAC)

MAC security policies govern the access on the basis of the classifications of subjects and objects in the system [6]. Objects are the passive entries storing information for example relations, tuples in a relation etc. Subjects are active entities that access the objects, usually, active processes operating on behalf of users.

An access class consists of two components: a security level and a set of categories.

The security level is an element of a hierarchically ordered set. The levels often considered are Top Secret (TS), Secret (S), Confidential (C) and Unclassified (U), where TS>S>C>U.

The set of categories is an unordered set, for example, NATO, Nuclear, Army etc.

The security level of the access class associated with an object reflects the sensitivity of the information contained in the object which means the potential damage which could result from unauthorized disclosure of information [7]. The security level of the access class associated with a user is called clearance, which reflects the user's trustworthiness not to disclose sensitive information to users not cleared to it.

Access control in mandatory protection systems is based on the following two principles:

- **No read-up/ Read down:** A subject can read only those objects whose access class is dominated by the access class of the subject.

- **No write-down/Write up:** A subject can write only those objects whose access class dominates the access class of the subject.

Satisfaction of these principles prevents information that is more sensitive to flow to objects at lower levels hence prevents the confidentiality of sensitive information. The effect of these rules can be diagrammatically represented as shown in fig. 5.

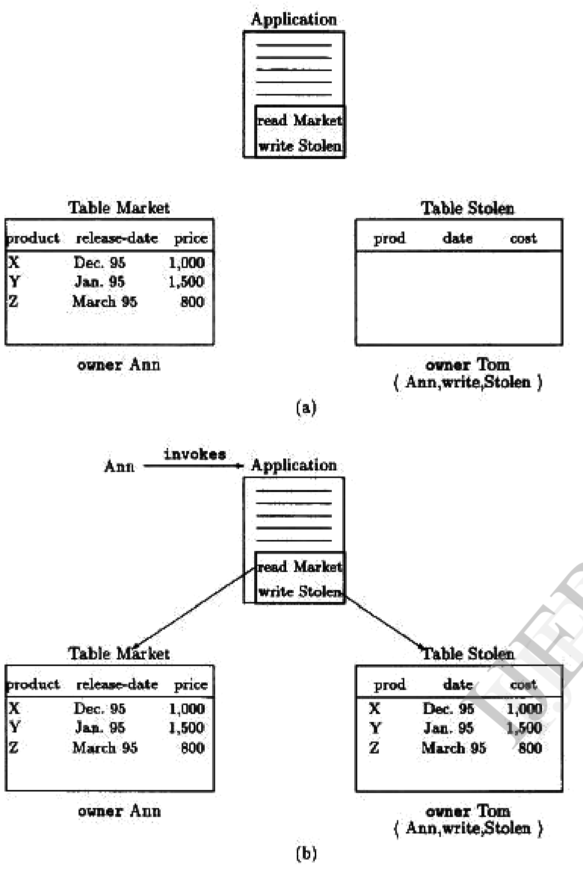

Fig 4: Example of Trojan horse

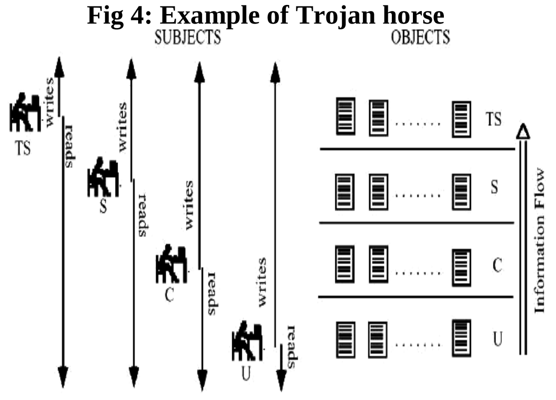

Fig 5: Controlling information flow for secrecy

MAC can as well be applied for the protection of information integrity [7]. For example, the integrity levels could be Crucial (C), Important (I) and Unknown (U). The integrity level associated with an object reflects the degree of trust that can be placed in the information stored in the object, and the potential damage that could result from unauthorized modification of the information. The integrity level associated with a user reflects the user's trustworthiness for inserting, modifying or deleting data programs at that level. Principles that are required to hold are as follows.

- **Read up** – A subject's integrity level must be dominated by the integrity level of the object being read.
- **Write down** – A subject's integrity level must dominate the integrity level of the object being written.

The effect of these rules can be diagrammatically represented as shown in fig. 6.

MAC models are not vulnerable to Trojan horse attacks: Consider fig. 4, if Tom is not allowed read access to table Market, under MAC control, table Market will have an access class that is either higher than or incomparable to the access class given to Tom. But then a subject able to read Market would not be able to write table Stolen and hence Trojan horse would not be able to complete its function.

5. ROLE - BASED ACCESS CONTROL (RBAC)

In, Role-based access control (RBAC), permissions are associated with roles, and users are made members of appropriate roles (fig. 7). This greatly simplifies management of permissions [16]. Roles are closely related to the concept of user groups in access control. However, a role brings together a set of users on one side and a set of permissions on the other, whereas user groups are typically defined as a set of users only.

5.1 RBAC Terminology

- **Objects** – Any system, resource file, printer, terminal, database record, etc.
- **Operations** - An operation is an executable image of a program, which upon invocation executes some function for the user.
- **Permissions** - Permission is an approval to perform an operation on one or more RBAC protected objects [10].
- **Role** - A role is a job function within the context of an organization with some associated semantics regarding the authority and responsibility conferred on the user assigned to the role.

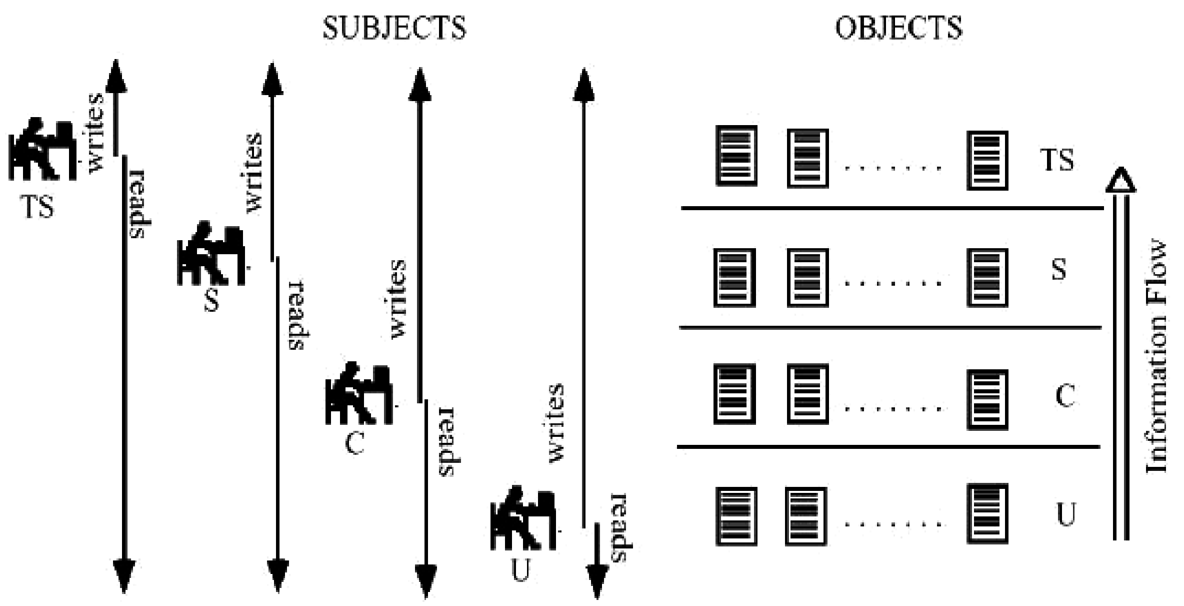

Fig 6: Controlling information flow for integrity

- **User** - A user is defined as a human being or a process executing on behalf of a user.
- **Group** - A set of users.
- Constraint – A relationship between or among roles.
- **Session** – A mapping between a user and an activated sub set of the set of roles the user is assigned to.
- **Role Hierarchy** – A partial order relationship established among roles.

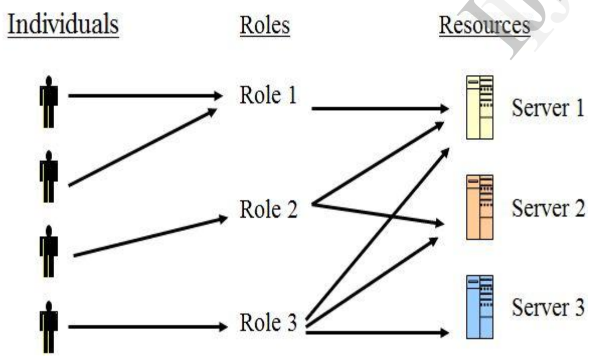

Fig 7: Role Based Access Control

5.2 Model Overview – Flat RBAC

Fig. 8 shows three sets of entities called Users (U), roles (R) and permissions [9]. A user in this model is a human being or other autonomous agent such as a process or a computer. A role is a job function or job title within the organization with some associated semantics regarding the authority and responsibility conferred on a member of the role. Permission is an approval of a particular mode of access to one or more objects in the system. Permissions are always positive and confer the ability to the holder of permission to perform some actions in the system..

Flat RBAC requires that user-role assignment (UA) and permission-role assignment (PA) are many-to-many relations. Flat RBAC requires support for user-role overview whereby it can be efficiently determined which roles a given user belongs to and which users a given role is assigned to.

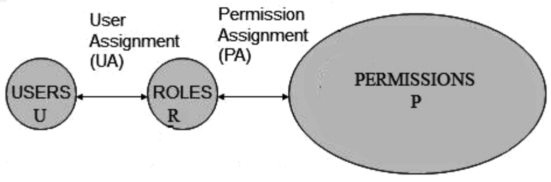

Fig 8: Flat RBAC

5.3 Hierarchical RBAC

The fig. 9 represents the Hierarchical RBAC which differs from fig. 8 only in introduction of the role hierarchy relation.

Role Hierarchy: Role hierarchies are a natural means for structuring roles to reflect an organisation's lines of authority and responsibility [11] [12]. These hierarchies are partial orders.

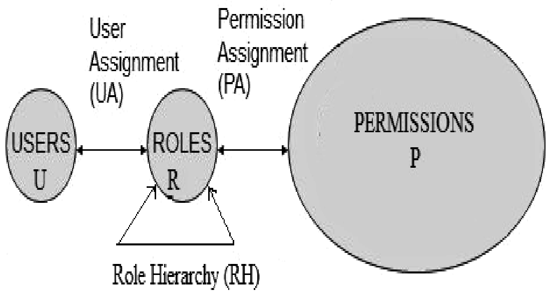

Fig 9: Hierarchical RBAC

A partial order is a reflexive, transitive and anti-symmetric relation. More powerful (or senior) roles are shown toward the top of role –hierarchy diagrams, and less powerful (or junior) roles towards the bottom.

- **General Hierarchical RBAC**

In this case there is a support for an arbitrary partial order to serve as the role hierarchy. The diagram below (fig. 10) shows a general hierarchy that facilitates both sharing and aggregation. Within this there is a junior-most role engineering department and senior most role Director. In between there are roles for two projects. Each project has a senior most role project lead 1 and 2 respectively and junior most role Engineer 1 and 2 respectively. In between each project has two incomparable roles production engineer and quality engineer for project 1 and 2 both.

- **Limited Hierarchical RBAC**

If any restriction is imposed on the structure of the role hierarchy then we are in this case. In this case Hierarchies are limited to simple structures such as trees or inverted trees.

Fig. 11 shows a tree hierarchy in which senior roles aggregate the permission of junior roles. Thus, PL1 acquires the permission of PE1 and QE1, and may have additional permissions of its own. Trees are good for aggregation but do not support sharing.

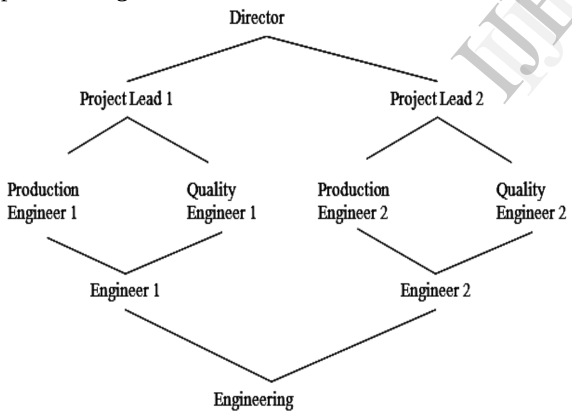

Fig 10: A General Hierarchy

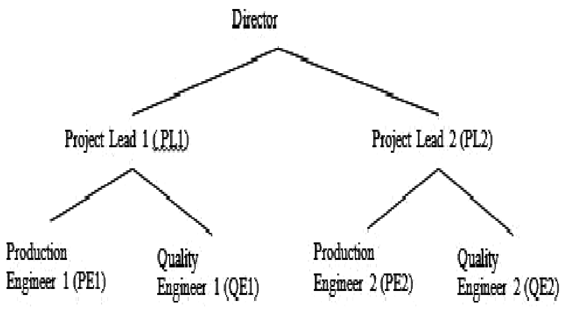

Fig 11: A Tree Hierarchy

Fig. 12 shows an inverted tree hierarchy. There is a junior – most role ED to which all employees in the department belong. Senior to this role are roles for two projects within the department, project 1 on left and project 2 on the right. Each project has an engineer role senior to it. These are production and quality engineer roles. An inverted tree facilitates sharing of resources.

5.4 Constrained RBAC

Constrained RBAC adds constraints to the hierarchical RBAC model. Constraints may be associated with the user-role assignment (static, fig. 13), or with the activation of roles within user sessions (dynamic, fig. 14).

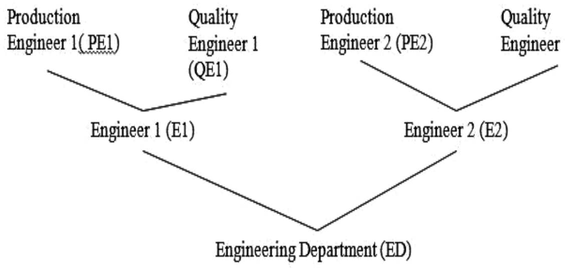
Production Engineer 1(PE1)
Quality Engineer 1 (QE1)
Production Engineer 2 (PE2)
Quality Engineer
Engineer 1 (E1)
Engineer 2 (E2)
Engineering Department (ED)

Fig 12: An inverted tree Hierarchy

Separation requirements are used to enforce conflict of interest policies that organizations may employ to prevent users from exceeding a reasonable level of authority for their positions.

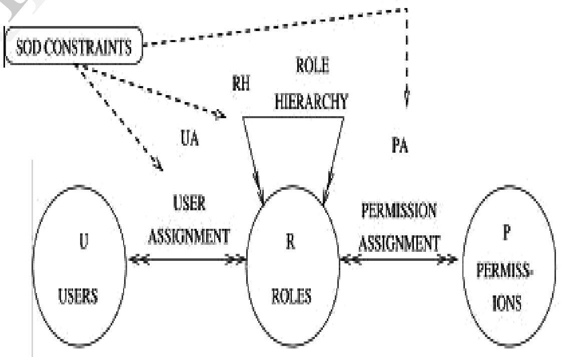

Fig 13: Constrained RBAC – Static SOD

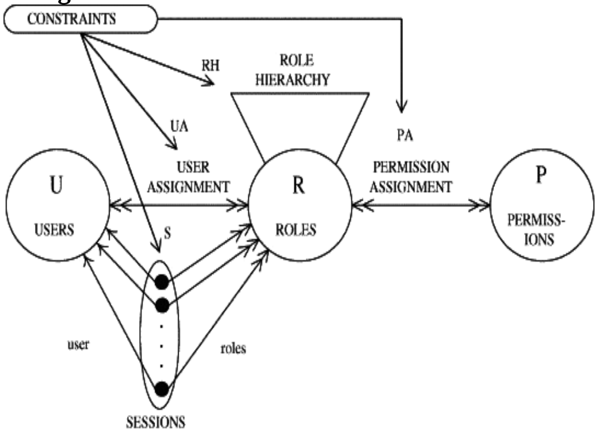

Fig 14: Constrained RBAC – Dynamic SOD

Separation of duty refers to the partitioning of tasks and associated privileges among different roles so as to prevent a single user from gathering too much authority. Separation of duty is of two types:

- **Static Separation of Duty**

The static separation of duty is also called strong exclusive which states that "A principal may not be member of any two exclusive roles" [13]. It means that the user is authorized of one role may not be authorized of another role or two roles have no any shared principle.

- **Dynamic Separation of Duty**

The Dynamic separation of duty is also called weak exclusive which states that "A principal may be a member of any two exclusive roles but must not activate them both at the same time" [13]. So, the user is authorized of both roles but both roles cannot be active at the same time.

- **Principle of Least Privilege**

Within RBAC system separation concepts are supported by the principle of least privilege.

The principle of least privilege requires that a user be given no more privilege than necessary to perform a job [13]. Ensuring least privilege requires identifying what the user's job is, determining the minimum set of privileges required to perform that job, and restricting the user to a domain with those privileges and nothing more.

6. Conclusion

Database security is an important goal of any data management system. Database security is based on three important constructs confidentiality, integrity and availability. Access control maintains a separation between users on one hand and various data and computing resources on the other. There are three main models of access control: DAC model govern the access to data on basis of user's identity predefined rules but are vulnerable to Trojan horse attacks. MAC models govern the access by users to data on the basis of classifications assigned to them but are too rigid to some environments. In RBAC models regulate access on the basis of roles user play in a system. Roles can be defined as set of actions or responsibilities associated with a particular working activity. RBAC makes the management of permissions easier, supports for principle of least privilege, separation of duties etc. All the models are not perfect and have their own vulnerabilities thus must be chosen according to the needs of organizations.

7. References

[1] E. Bertino and R. Sandhu, "Database Security – Concepts, Approaches and Challenges", IEEE Transactions on Dependable and Secure Computing, Volume 2 , No. 1, January- March ,2005, pp. 2-16.

[2] Meg Coffin Murray, "Database Security: What students need to know", Journal of Information Technology Education: Innovations in Practice, Volume 9, 2010, pp. 61-77.

[3] A. B. Dastjerdi, J. Pieprzyk, and R. S. Naini. , "Security in Database: A Survey Study", Department of Computer Science, The University of Wollongong, 1996, pp. 1- 39.

[4] Emil Burtescu, "Database Security- Attacks and Control Methods", Journal of Applied Quantitative Methods, Volume 4, No. 4, 2009, pp. 449-454.

[5] E. Bertino, S. Jajodia and P. Samarati, "Database Security: Research and Practice", Pergamon, Information Systems, Volume 20, No. 7, 1995, pp. 537-556.

[6] R. Dollinger, "Database Security Models- A Case Study", European Symposium on Research in Computer Security, 1990, pp. 35-41.

[7] R. Sandhu and P. Samarati, "Access Control: Principles and Practice", IEEE, Communications Magazine, 1994, pp. 40-48.

[8] R. A. Crues, "Methods for Access Control: Advances and Limitations", Harvey Mudd College, Claremont, California.

[9] Sandhu, Ravi, David Ferraiolo, and Richard Kuhn, "The NIST model for role-based access control: Towards a unified standard", Symposium on Access Control Models and Technologies, Proceedings of the fifth ACM workshop on Role-based access control, Volume 26, No. 28, 2000, pp. 47- 63.

[10] R. Sandhu and E. J. Coyne, "Role-Based Access Control Models", IEEE Computer, 1996, pp. 38-47.

[11] R. Sandhu, E. J. Coyne H. L. Feinstein and C. E. Youman, "Role – Based Access Control Models", IEEE Computer, Volume 29, No. 2, Februaryn1996, pp. 38-47.

[12] S. L. Osborn, "Role- Based Access Control", Springer, Security, Privacy, and Trust in Modern Data Management, 2007, pp. 55-70.

[13] D. F. Ferraiolo and R. Kuhn, "Role – Based Access Control", 15th National Computer Security Conference, Baltimore, 1992, pp. 554-563.

[14] Huwida., M. Guimaraes, Z. Maamar, and L. Jololian, "Database and Database Application Security", ACM SIGCSE Bulletin, Volume 41, No. 3, 2009, pp. 90-93.

[15] S. Imran, Dr. I. Hyder, "Security Issues in Databases", IEEE, Second International Conference on Future Information Technology and Management Engineering, 2009, pp. 541-545.

[16] J. Zheng, Q. Zhang, S. Zheng and Y. Tan, "Dynamic Role-Based Access Control Model",

JOURNAL OF SOFTWARE, Volume 6, NO. 6, JUNE 2011, pp. 1096-1102.

[17] Li N., J. Byun, and E. Bertino, "A critique of the ANSI standard on role-based access control", Security & Privacy, IEEE Volume. 5, no. 6, 2007, pp. 41-49.

[18] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D. Richard Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control", ACM Transactions on Information and System Security (TISSEC), Volume 4, No. 3, 2001, pp. 224-274.

[19] T. F. Lunt and E. B. Fernandez, "Database security", ACM SIGMOD Record 19, No. 4, 1990, pp. 90-97.