[VNSGU JOURNAL OF SCIENCE AND TECHNOLOGY] Vol.4. No. 1,July, 2015
67 - 76,ISSN : 0975-5446

Survey of Access Control Models for its Applicability In Object Oriented Database Security

DINDOLIWALA Vaishali J.
Assistant Professor,
C. B. Patel Computer College,
Bharthana, Vesu, Surat
vaishali_1331@yahoo.co.in

MORENA Rustom. D.
Professor,
Department Of Computer Science,
Veer Narmad South Gujarat Uni., Surat
rdmorena@rediffmail.com

Abstract

In database security, the major focus is on access control of the different legitimate users. Object Oriented Database security is a mechanism that controls access to and use of the database at the object level. The various database security policies determines which users have access to a specific schema object and the actions that can be performed by each user on the specified object. Most of the existing access control models, for instance Discretionary Access Control (DAC), Mandatory Access Control (MAC) or Role-Based Access Control (RBAC) have been originally designed for relational database systems. However, the application of these models in object oriented systems can't be straight forward, due to object-oriented characteristics like object identity, encapsulation, inheritance, polymorphism and complex objects which requires the integration of new mechanisms to legacy access control concepts.

This paper surveys different access models used for database security, compares them and explores advantages, disadvantages and limitation if any of each of them along with the access control mechanism which could be preferably applied in Object Oriented Databases.

Keywords: Access Control**,** DAC, MAC, RBAC, Object-Oriented Database

1. Introduction

Object Oriented Database is the integration of various database capabilities like integrity, persistency, security etc. with object oriented programming language capabilities like encapsulation, inheritance, polymorphism, object identity etc.

In Object Oriented Databases, user can create classes. Each class has unique identifier. Class and its objects are persisted. Each object of a class has unique object identifier. Depending upon the

access rights, different users can view class objects or can modify objects attributes. Also it is necessary to protect the database from unauthorized access.

For providing database security, several access control models are used. The main objective of these models is to provide a verifiable system which guarantees the protection of information being accessed by unauthorized access. The access control models elaborate generic security objectives of secrecy, integrity and availability, in the context of a particular system.

2. Access Control Models

Access control is a collection of methods and components used to protect information assets. Some information from the database can be accessible by everyone. But there is a need to restrict access of other information of the database. Access control supports the confidentiality and the integrity properties of a secure system. The confidentiality property protects information from unauthorized expose. And the integrity property protects information from unauthorized modification [4]. Thus Access control ensures that only authorized users can view database information. Access control models are usually used to manage how information can be accessed and shared on a system. The most commonly used access control models have been discussed in this paper

.

2.1 Mandatory Access Control Policy

The main elements of this model are:

- **Objects** - passive entities containing information to be protected.
- **Subjects -** active entities requiring accesses to objects. E.g., users, processes etc.
- **Access modes**: types of operations performed by subjects on objects. These operations may be read, write, append etc.

MAC policy secures information by assigning sensitivity levels or labels to each subject and object. Some examples of sensitivity levels are public, sensitive and secret. A subject's security label defines the security clearance or category of object that the subject is permitted to access. A label on a data item is called a security classification, while a label on a user is called a security clearance [2, 3]. Based on security labels, it determines whether an access request should be granted or denied. Sometimes it is also called as Rule Based Access Control. A MAC policy is more secure and is usually used in environments where confidentiality is most important, such as a military institution [5].

When a user attempts to access a resource under MAC, the system checks the user's classification and categories and compares them to the properties of the object's security label. If the user's credentials match the MAC security label properties of the object, then the access is allowed. For example, if a user has a security clearance of secret and he requests a data object with a security classification of top secret, then the user will be denied access because his clearance is lower than the classification of the object. Here, it is necessary that both the classification and categories must match. A user with top secret classification, for example, cannot access a resource if they are not also a member of one of the required categories for that object.

2.1.1 Benefits

MAC is relatively straightforward and is considered a good model for commercial systems that operate in hostile environments where the risk of attack is very high, confidentiality is a primary access control concern or the objects being protected are valuable. So, it is the main access

control model used by the military and intelligence agencies to maintain classification policy access restrictions.

2.1.2 Limitations

MAC systems are difficult and expensive to implement due to the reliance on trusted components and the necessity for applications to be rewritten to adhere to MAC labels and properties. With mandatory controls, only administrators and not owners of resources may make decisions that bear on or derive from policy [6]. The assignment and enforcement of security levels by the system under the MAC model places restrictions on user actions that, while adhering to security policies, prevents dynamic alteration of the underlying policies and requires large parts of the operating system and associated utilities to be trusted and placed outside of the access control framework.

It also does not provide dynamic separation of duty. Under a MAC enforced environment access to all resource objects such as data files is controlled by settings defined by the system administrator. As such, all access to resource objects is strictly controlled by the operating system based on system administrator configured settings, it is not possible under MAC enforcement for users to change the access control of a resource. MAC requires a considerable amount of planning before it can be effectively implemented. Once implemented it also imposes a high system management overhead due to the need to constantly update object and account labels to accommodate new data, new users and changes in the categorization and classification of existing users [20].

2.2 Discretionary Access Control Model

The main elements of this model are:

- **Objects –** Information need to be protected.
- **Subjects –** An entity that can access objects.
- **Privileges** – An action that can be performed on objects by the subjects.

In DAC Policy, access to objects is restricted based on the authorizations granted by the owner to the user. The owner of the object specifies which subjects can access the object. Users can be given the ability of passing on their privileges to other users, where granting and revocation of privileges is regulated by an administrative policy [1, 7, 18]. Owner of the objects can grant permissions to other users also.

2.2.1 Benefits

Any database management system must enforce DAC policy allowing users to specify and control sharing by named individuals, groups of individuals or by both which limits propagation of access rights and includes or excludes access to the granularity of a single user. The database system that uses DAC has the ability for the owner of an object to assign or revoke rights to view or modify the object.

2.2.2 Limitation

The permissions that the end user has are inherited into other programs they execute. This means the end user can execute malware without knowing it and the malware could take advantage of the potentially high level privileges the end user possesses. Also, for each user, the required privileges have to be set explicitly.

2.3 Role Based Access Control Model

One of the weaknesses of DAC is that it does not facilitate management of access rights. Each user must be explicitly granted every privilege that they need to accomplish their tasks. Sometimes it may happen groups of users need similar privileges. For example, all Doctors in a Hospital might require identical privileges; all Nurses might require identical privileges, which are different from those of the Doctors. So, instead of assigning each privilege to users individually, roles are created. Each role is assigned the required privileges according to their responsibilities, qualification and specification of duties in an organization. And then the role is assigned to users [3, 8]. RBAC policy allows the creation of roles for Doctor and Nurse. Roles can also apply to military systems; for example, target analyst, situation analyst and traffic analyst are common roles in tactical systems. RBAC is in fact a form of mandatory access control, but it is not based on multilevel security requirements [21].

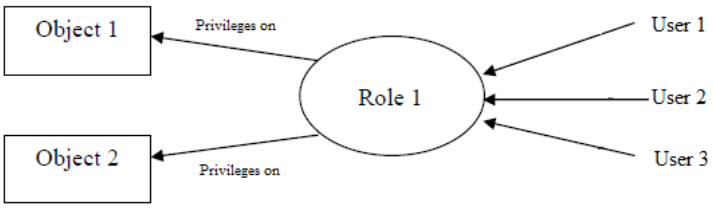

Figure 1 Role Terminology

2.3.1 Benefits

RBAC enforces separation of duties by the administrator which minimizes fraud transactions. In real situations, only certain transactions need to be restricted under separation of duty requirements [1, 7, 9, 10]. Once the transactions of a Role are established within a system, these transactions tend to remain relatively constant or change slowly over time. When a new person enters the organization, the administrator simply grants membership to an existing role. When a person's function changes within the organization, the user membership to his existing roles can be easily deleted and new one can be easily granted. Finally, when a person leaves the organization, all memberships to all Roles are deleted. Also, RBAC is to support integrity [1]. Least Privilege is supported by the RBAC because RBAC can be configured so only those permissions required for tasks conducted by members of the role are assigned to role. It is a nondiscretionary access control mechanism which allows and promotes the central administration of an organizational specific security policy. As the number of users/objects increases in a system, it would lead to scalability problems with the traditional DAC and MAC models for access control which will be overcome by RBAC. With RBAC it becomes easier to reassign users from one role to another, or to alter the privileges for an existing role. It also enforces constraint management. The constraints can also be dynamic. A constraint like at most one user at a time can activate the role can be handled [7].

2.3.2 Limitations

In RBAC, there is a lack of static checking. It is difficult for programmers to ensure that their code properly manages the access control policy. Programmers must manually keep track of what roles must be held when each function is invoked, which depends on the set of privileged operations that can be reached during the function's execution. If a function is ever executed in the wrong environment, there will be a runtime role failure which is difficult to diagnose and fix.

From the above discussion of DAC, MAC and RBAC, the comparison among them is shown in following Table 1

.

Table 1 Comparison among DAC, MAC and RBAC

Characteristics	DAC	MAC	RBAC
Who can decide permissions/policies?	Owner	Administrator	Administrator
Who can assign security controls?	Owner / User	Administrator	Administrator
It is based on?	personal permissions	Level of authorization/ clearance of the accessing object	“group" - level permissions
Definition of permissions	Dynamic	Static	Static
Each resource has Security maintained by	ACL	Security Label	Role
Administration Cost?	Increased because explicit permissions for each user have to be set.	Increased because harder to implement and often require consideration adapting to ensure all applications function correctly.	Reduced because permissions are granted to many users simultaneously by creating roles. Roles can be updated without updating the privileges for every user on an individual base.
Who can revoke Access Revocation?	Anyone with authority to grant also has authority to revoke.	Administrator	Role will be revoked from user by administrator
Used for?	operating systems and relational database systems	military or sensitive national security information	Commercial sectors
Enabling Separation of Duties?	No	No	Yes
Can user be assigned to more groups?	No, there is no concept of group.	No, there is no concept of group.	Yes.
Provides central control over data?	No	Yes	Yes
Can individual set/change the access permission?	Yes	No	No
Can owner transfer authenticated objects or information access to other	Yes	No	No

users?			
Is it Policy-neutral?	No	No	Yes
Change of Ownership possible?	Yes	No	No

3. Literature Survey

At present no standard data model for object-oriented systems has been proposed. Until one becomes available, the choice of an object-oriented data model is an arbitrary one that is determined by the nature of the users of the Object Oriented Database Management System.
Jonathan K. Millen and Teresa F. Lunt [15] have proposed a label-based reference-monitor model which uses the concept of MAC. Each object has a single security level that applies to everything within it which is analogous to tuple-level labeling in the relational model. If a subject wants to read or write variables to some other object, it has to send a message to request retrieval or modification of some variable there. Because of inheritance, a subject may implicitly read variables or methods in objects above it in the hierarchy.

Eduardo B. Fernandez, Ehud Gudes and Haiyan Song [16] have developed an authorization model for object oriented databases. They have provided the security at class level. They have provided 3 policies: a user that has access to a class is allowed to have similar type of access in the corresponding subclasses to the attributes inherited from that class, access to a complete class implies access to the attributes defined in that class as well as to attributes inherited from a higher class and an attribute defined for a subclass is not accessible by accessing any of its superclasses.
M.B.Thuraisingham [17] has proposed a multilevel secure object-oriented data model SORION in which mandatory access control policy is used for classes, class objects, class methods and instance variables. Here subjects and objects are given security labels.

T. F. Keefe, W. T. Tsai and M. B. Thuraisingham [19] have proposed multilevel security model for object oriented database called SODA. In this approach each object is given a classification. Objects are labeled with their sensitivity levels. Subjects are associated with clearance levels. Mandatory security is enforced with a combination of compile-time and run-time checks. A multilevel secure computer arbitrates all access of objects by subjects. The arbitration is carried out by the reference monitor according to a security policy. Each instance variables also have sensitivity level.

Premchand B. Ambhore, B.B.Meshram and V.B.Waghmare [11] have proposed two DAC models for OODBMS. One is the DAMKLES access control models and the other is the LPGSF access control model. In first model, every object is further broken down into smaller access units called protection objects. The access control allows users to grant a privilege to other users if user is the owner of the object. The second model is based on a set of policies that define authorization inheritance through class hierarchies.

Jeffrey Fischer, Daniel Marino, Rupak Majumdar and Todd Millstein [12] have presented the design, implementation, formalization and practical validation of Object-sensitive RBAC (ORBAC), a generalization of the widely used RBAC model for access control. In this model, they have created users. Each user has some roles assigned. They have specified which class method will be accessed by each roles. Methods are protected by assigning required privileges through annotations in the code. Only coarse-grained policies, which restrict access at the level of classes rather than objects, are supported. For each role, they have also saved which parameters

are required. So the limitation of this model is - role parameters do not change. If role parameters can change, the type system becomes unsound, potentially allowing prohibited calls.

4. Security Policy Related to Object Oriented Databases

Different models have been used for implementation of security in object oriented databases. Each security model has to make some assumptions about the object oriented model used for its particular database. These models differ on many aspects because they focus on different aspects of the security problem or because they make different assumptions about what constitutes a secure database.

Inheritance is the fundamental property of object-oriented systems which often goes together with associations and aggregation. A class object may have many descendants or instances. Using DAC, it is difficult to maintain access to a class. If a user is authorized to access a class, then that user should also be authorized to access all of its instances, hence that user may have authorization on inherited attributes also. If MAC is used, then each object may be assigned some security label and each user may be assigned some security clearance label. Each time user is going to access objects, his clearance label should be checked with objects' security label and then access will be granted or denied which is difficult to handle. Also it may happen that user has different authorization on these classes. Apart from this, the object-oriented database model also permits the classification of an object's sensitivity through the use of class and instance. When an instance of a class is created, the object can automatically inherit the level of sensitivity of the superclass. This prevents a flow control problem, where information passes from higher to lower classification levels [18].

Another important property of object-oriented systems is encapsulation that manages the state (instance variables) and behavior (methods) of an object. Methods perform all reading and writing of the data in an object. For this reason, the data is encapsulated in the object. This is one of the important differences between object-oriented and relational databases. All control for access, modification, and integrity start at the object level. Only methods can change the state of the object. One may not manipulate or view an object's hidden data; instead one sends a message to an object and the object itself selects the method by which it will react to the message [14]. So there is a need to provide proper authorization for method invocation. If DAC is used, then with each method there is a need to maintain authorization list which tells who can access that method. If MAC is applied, then access to an object is controlled by checking security level of the object, which is computed according to the security rules of the system.

Once class is created in an object oriented systems, it is necessary to restrict creation of class instances. It is necessary to specify who will create, modify or delete an instance of a class. In most access control models, class level privileges are assigned. In object oriented systems, a user can create any number of objects of a class. He can decide who can access those objects. Owner of the objects can give privileges to other users to access those objects. So instead of assigning privileges to class, it is more suitable to assign privileges for class methods. If there is a class based authorization, anyone can access all instances of a class if he has authorization for that class. So any instance information can be accessed. If there is method based authorization then user must have authorization on that method to access the object. Also, the security mechanism should not only consider the permission of attributes of objects but also the permission of methods [13]. So in this case, DAC or MAC is inappropriate choice. If RBAC is used as security model, administrator can make a role which has privileges defined in it for class methods and then assign it to all the users who have similar access control which also reduces administrative work.

Also in object oriented systems, users can define class members - public, private or protected. If methods are public, they are open for all users. While in private methods, users can keep basic commands themselves away from other users. For example, if there is a class called BankAccount which has variables like AccountNo and Balance. If these member variables are public then any object in the system can change the Balance to zero or some negative value which could cause the program to fall over. Instead of this, user will have provided a getBalance() and setBalance() method and can make variable Balance private or protected member. And within these methods, user can handle the Balance variable. So that other objects can still access the data, but they can't put invalid data in. That means other external object can't have direct access on Balance but Balance can be updated through the proper method invocation. If DAC is used for method authorization, then user can grant method authorization to other user also in which data integrity will not be maintained.

The comparison of DAC, MAC and RBAC with respect to object oriented characteristics is shown in following Table 2.

Table 2. Comparison among DAC, MAC, RBAC based on various characteristics

Characteristics	DAC	MAC	RBAC
Maintenance of inheritance	Good	Better, but requires more efforts.	Best
Maintenance of Encapsulation	Good	Better, but requires more efforts.	Best
Can Data Integrity be maintained?	No	Yes	Yes
Can Consistency be maintained?	No	Yes	Yes
Can Information flow be maintained?	No	Yes	Yes
Can a malicious program change it? Or Can illegal access possible?	Yes	No	No

5. Methodology for Access Controls in Object Oriented Database

As discussed above, RBAC will be more suitable to implement security in Object Oriented Database. Using roles, privileges will be assigned to the users based on the user's job requirements.

In object oriented databases, rather than assigning permissions for each attribute of a class, it is more convenient to assign permission on class methods from which the access to class attributes can be controlled. In Object Oriented Systems, resources are objects or attributes of objects and access privileges are objects methods. Thus, granting an access privilege to an object consists in authorizing the object method call.

Another way to provide security of data is encryption. Without database encryption, it makes no sense to guarantee that the sensitive information will be protected against untrustworthy users. It can be employed to maintain data integrity ensuring that even a little modification made on the data can be detected. Database encryption technology meets the data confidentiality requirements and has become an indispensable aspect of enterprise database security.

6. Conclusion

A role is a group of users that have the same job functionality within an organization. Roles access objects based on role rights. The object is concerned with the user's role but not the user. Users frequently change but the roles are not, which makes RBAC a better access control mechanism than DAC and MAC. Implementing RBAC in existing object oriented databases comes with its challenges. It takes time and effort to determine the permissions each role will be assigned. But rather than assigning each access rights on methods to users individually, it is better to assign access rights on methods to roles and then assign roles to the different users. Hence, RBAC is more suitable as an access control model for an object oriented databases.

References

[1] David F. Ferraiolo, D. Richard Kuhn, Role-Based Access Controls, 15th National Computer Security Conference, Baltimore, (1992), 554 – 563.

[2] Ravi S. Sandhu, Sushil Jajodia, Data And Database Security And Controls, Handbook of Information Security Management, Auerbach Publishers, (1993), 481-499.

[3] Ravi S. Sandhu, Relational Database Access Controls, Handbook of Information Security Management, Auerbach Publishers, (1994), 145-160.

[4] Shabnam Mohammad Hasani, Nasser Modiri, Criteria Specifications for the Comparison and Evaluation of Access Control Models, I. J. Computer Network and Information Security, 5 (2013), 19-29.

[5] Walid Rjaibi, Paul Bird, A Multi-Purpose Implementation of Mandatory Access Control in Relational Database Management Systems, Proceedings of the 30th VLDB Conference, Toronto, Canada, (2004), 1010-1020.

[6] Steve Demurjian, Implementation of Mandatory Access Control in Role-based Security System with Oracle Snapshot Skill, CSE 367 Independent Study Final Project Report, Computer Science & Engineering, the University of Connecticut Storrs, CT 06269-3155, December, 13 2001.

[7] Pierangela Samarati, Sabrina de Capitani di Vimercati, Access Control: Policies, Models and Mechanisms, LNCS, 2171 (2001), 137–196.

[8] Ji-Won Byun, Ninghui Li, Purpose Based Access Control for Privacy Protection in Relational Database Systems, The VLDB Journal — The International Journal on Very Large Data Bases, Volume 17 Issue 4 (2008), 603-619.

[9] Sumitro Bhaumik, Jyotishkar Dey, Distributed Database Security using Discretionary Access Control, a project report, faculty of Engineering and Technology, Jadavpur University, 2009-2013.

[10] Wolfgang Essmayr, Stefan Probst, Edgar Weippl, Role-Based Access Controls: Status, Dissemination and Prospects for Generic Security Mechanisms, Kluwer Academic Publishers. Manufactured in the Netherlands., Electronic Commerce Research 4,© 2004, 127-156.

[11] Premchand B. Ambhore, B.B.Meshram, V.B.Waghmare, An Implementation of Object Oriented Database Security, Fifth International Conference on Software Engineering Research, Management and Applications, IEEE Computer Society, ISBN : 0-7695-2867-8 (2007), 359-365.

[12] Jeffrey Fischer., Daniel Marino, Rupak Majumdar, Todd Millstein, Fine-Grained Access Control with Object-Sensitive Roles, European Conference on Object-Oriented Programming, Genova, Italy, July 6-10, 2009.
[13] Haibo Hu, Hong Xiang, Implicit Authorization Mechanism of Object-Oriented Database, World Academy of Science, Engineering and Technology, Volumn 4, No. 2 (2010), 561-566.
[14] Oscar Nierstrasz, A Survey of Object-Oriented Concepts, Object-oriented concepts, databases, and applications, ISBN : 0-201-14410-7 (1989), 3-21.
[15] Jonathan K. Millen, Teresa F. Lunt, Security for Object-Oriented Database Systems, Research in Security and Privacy Proceedings, IEEE Computer Society, ISBN : 0-8186-2825-1 (1992), 260-272**.**
[16] Eduardo B. Fernandez, Ehud Gudes, Haiyan Song, A Security Model For Object- Oriented Databases, IEEE Symposium on Security and Privacy Proceedings, ISBN : 0-8186-1939-2 (1989), 110-115.
[17] M.B.Thuraisingham, Mandatory Security In Object-Oriented Database Systems, OOPSLA '89 Proceedings, ACM 089791-333-7 (1989), 203-210.
[18] James Cannady, Security Models For Object-Oriented databases, Auerbach Publications, CRC Press LLC, 82-10-44,© 1997.
[19] T. F. Keefe, W. T. Tsai, M. B. Thuraisingham, SODA: A Secure Object-oriented Database System, Computers and Security Volumn 8, No. 6, (1989), 517-533.

URLs

[20] http://www.techotopia.com/index.php/Mandatory,_Discretionary,_Role_and_Rule_Based_Access_Control
[21] shttp://www.networkcomputing.com/showarticle.jhtml