World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:4, No:2, 2010

Implicit Authorization Mechanism of Object-Oriented Database

Haibo Hu, and Hong Xiang

***Abstract*—**Due to its special data structure and manipulative principle, Object-Oriented Database (OODB) has a particular security protection and authorization methods. This paper first introduces the features of security mechanism about OODB, and then talked about authorization checking process of OODB. Implicit authorization mechanism is based on the subject hierarchies, object hierarchies and access hierarchies of the security authorization modes, and simplifies the authorization mode. In addition, to combine with other authorization mechanisms, implicit authorization can make protection on the authorization of OODB expediently and effectively.

***Keywords*—**Object-oriented database(OODB); security protection; authorization mechanism; implicit authorization; authorization check.

I. Introduction

WITH the growing development of computer network technology, database systems have to face the resulting requirements of data centralization and multi-user demand for multi-service access. Database security issues have become increasingly complex and serious [1]. OODB introduced the concept of object-oriented into the database system to make a significant progress in the history of database system's development [2]. Since the generation of the Object-Oriented Database, its security issues have always attracted extensive research and attention of vast number of scholars. With the wide applications of OODB in various fields, the researches on its security mechanism is also with great significance.

In the researches of security of OODB, the authorization mechanism and access control have always been the contents of which have been noted. 1994, Fernandez and others put forward the concept of authorization strategy with a class hierarchy, and further improved the five dimensions which have been widely applied in the relational database to three dimensions [3]. Subsequently, Bertino [4] and others added the effectiveness time limits into three dimensions and raised a temporary authorization model (Temporal Authorization), but its effectiveness and flexibility is not enough. Demurjian [5] and others have tried applying the user-role security in the OO model. Thomas and others have modeling the multi-level security of OODB based on a hierarchy of the subject, and advanced the message transferring mechanism between the subject levels, this improved the researches to a new height [6]. Bertino [7] then combined strong authorization and weak authorization to improve the flexibility, and called for the concerted effort of research**.**

Based on comparing and summarizing the advantages and disadvantages of each model, this paper further developed the Subject, Object and access Operation hierarchy structures of OODB, and also expanded the implicit authorization to Subject and Operation levels, focusing on the study of implicit authorization control strategy.

Haibo Hu is with Chongqing University, 400030 P.R. China (phone: 86-23-65111025; fax: 86-23-65111025; e-mail: hbhu@cqu.edu.cn).

Hong Xiang is with Chongqing University, 400030 P.R. China (e-mail: xianghong@lcqu.edu.cn).

This work is supported by the National High Tech Research and Development Plan of China under grant No. 2007AA01Z445, and the Natural Science Foundation in Chongqing City (CSTC) of China under Grant No. 2008BB2312.

II. Features of OODB Security Mechanism

The special data model structure of object-oriented database determined its security mechanisms of the following characteristics:

A. Complexity of Objects

Different from the traditional relationship between the tables of RDB, the basic object-oriented database access unit is the object. Each object has its own unique identifier. Each object is unique and independence in the database. In addition, Object-Oriented data model is composed by the class hierarchy. There exist close correlation between class and sub-classes, class and instances. This makes the object hierarchy and the relationship between objects in the security mechanisms become more complicated.

B. Variety of Operation

The relationship in the relational model is only composed by data attributes. While the object (or class) in the OO data model is encapsulated by common attributes and methods [8]. The operation of OODB is not only limited to traditional additions, deletions, search and update. Users can also define user-methods based on actual requirements, and achieve the various operations on the DB through implementation of these methods. Therefore, the security mechanism should not only consider the permission of attributes of data but also the permission of methods.

C. Security of Original

Object-oriented data model itself has a certain degree of security protection measures. Since the encapsulation characteristics of Object-Oriented model, the state and behavior of OODB was encapsulated in respective object. Users implement the user-method or system-method though sends the

specified explicit messages to accomplish access or call [9]. Object-oriented data model itself promotes the data encapsulation and information transfer mechanism has also provided implementation restrictions and access protection of attributes and methods though public/private type.

D. Classified Protection

Object-oriented database provides a number of different security mechanism [10], such as: sub-graph mechanism, view mechanism, authorization mechanism, object model mechanism and so on, to realize the OODB security protection on the database level, sub- graph level, view level, class level and object level.

Although the view mechanism and model mechanism of OODB has , to some extent, provided a good protection, but did not provide the specified object to the specified user's access mechanism, that is unable to provide suitable accuracy.

OODB authorization mechanism based on the complexity hierarchy of object and operation, and combined strong/weak authorization and positive/negative authorization to complete the OODB access control flexibly and precisely.

III. Security Authorization Model

The authorization control model can be defined by a three-tuple (S, O, Op): access subject S, access Object O and access Operation Op. The relationship between these three elements can be illustrated in Fig. 1. If (S, O, Op) = true (or 1), it means that the subject S (commonly be user or control group) have the operation permission Op on object (or object set) O.

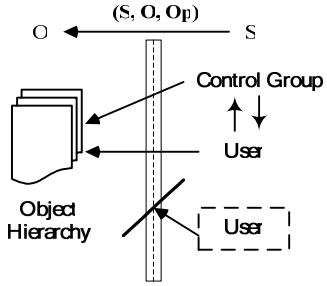

Fig. 1 Authorization Control Model

Based on the features of OODB security authorization model, we will make these extension and improvement on each element hierarchy of Authorization Control Model.

A. Subject Hierarchy

According to the role of user *u*, the object can be organized into different control group G_k .Access Object can be either user or group. The member of group can be either user or other group. These elements together construct an acyclic digraph. Access subject Hierarchy can be shown in the relationships between these user and control group. A control group can be composed by several users or control groups which hold the same access permission on the same subject, as "*u* directly belonged to G_k ": $u \in G_k$, or "*u* indirectly belonged to G_k": $u \in_1 G_1 \in_2 \ldots \in_n G_k$, show as Fig.2:

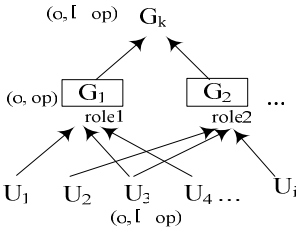

Fig. 2 Subject Hierarchy

B. Object Hierarchy

The division of the authorization Object level can effectively simplify the OODB database authorization set, and reduce the complexity of the authorization model. OODB object hierarchy includes all the authorization objects, and expanded the class method (Method) [11].

According to the granularity of authorization object, the object can be divided to different levels, from database system level to object attribute level. These object levels together composed a structure of acyclic digraph with root, shown as Fig. 3. The object authorization in higher level would include those of lower level implicitly.

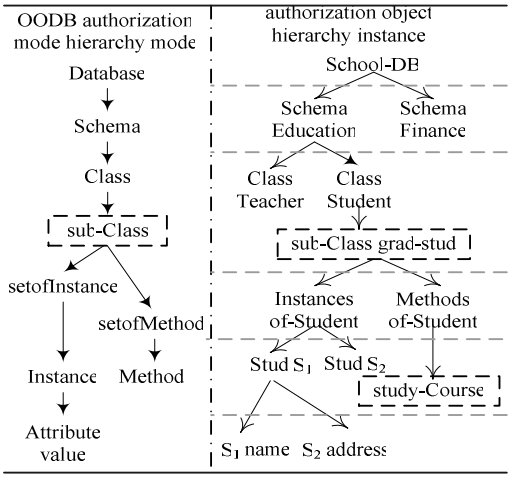

Fig. 3 Object authorization Hierarchy

C. Access Mode (Operation)

Access Mode to object of the subject also refers to the hierarchical associated relationship of the operation authority, which make the object has another operation implicitly when it has some authorization. In the associated relationship in access operation, written authorization implies read authorization (can be represented as `read<write`), and the authorization of new type definition implies the read authorization on definition (can be represented as `read<write`).

The hierarchies of OO Modal also include the authorization mechanism on methods. Methods include three operations in general, they are calling, modifying and creating, and the hierarchical relationships can be represented as:

`call<modify<create.`

IV. Authorization Control Mechanisms

In the hierarchical OODB Systems, authorization mechanism can be sufficient to support the various access requirements

flexibly [12]. It must make a unified control on the some authorizations which access subject precise and flexible, and also precisely positions the operation properties of different subject's authorization on a particular object hierarchy. Authorization mechanisms of OODB make full use of Positive Authorization and Negative Authorization (represented by "¬ ") to authorize and cancel the authorization.

A. Authorization Check

Once the subject requires accessing a certain object, firstly should perform the authorization check. According to the check result to decide whether the required access operation is allowed [13].

Assuming the subject S requires performing the access operation *Op* on object *O*, and then we should perform the permission check on subject S. The check result would direct whether the required operation is allowed or refused. According to the ideals of Bertino in citation [7], the basic thought of authorization check is from strong authorization to weak authorization, from positive authorization to negative authorization, to complete the authorization check on certain subject access permission, shown as Fig. 4:

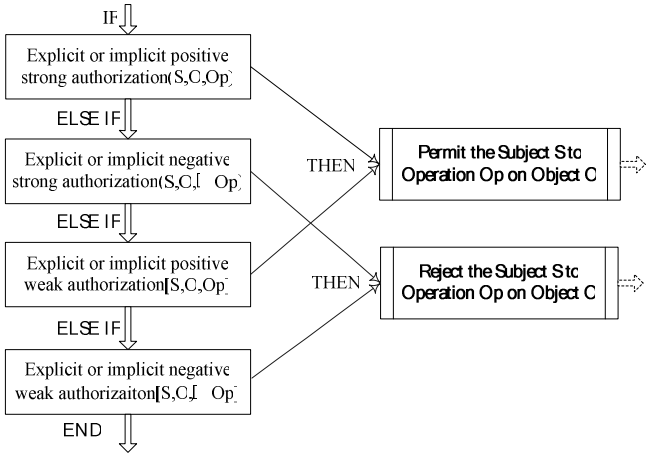

Fig. 4 Basic thought of authorization check

Based on the above mentioned check thought, on the subject hierarchy (Fig. 2), authorization check process begins from a certain user. According to the return result of check to decide whether terminate the whole check process or go on to check the above-level subject. Assume that user u requires the access operation *Op* on object *O*, detailed check process check_auth can be described as following, shown as Fig. 5:

(1) Firstly perform the strong authorization on user u:

Begin check_strong (u) :

```
if (exist(u,o,op)=1) then
   return true; exit check_auth;
elseif(exist (u,o,¬op)=1) then
   return false; exit check_auth;
else go to (2);
```

(2) Then perform the strong authorization on group G_i which user u directly belonged to, check_strong(G_i),$u \in G_i$:

```
if (exist(Gi,o,op)=1) then
   return true; exit check_auth;
elseif(exist (Gi,o,¬op)=1) then
   return false; exit check_auth;
else
   exit check_strong;
```

(3) And thengo on the weak authorization on user u:

Begin check_weak(u):

```
if ( exist [u,o,op]=1 ) then
   return true; exit check_auth;
elseif (exist [u,o,¬op]=1) then
   return false; exit check_auth;
else go to (4);
```

(4) Next, perform the weak authorization on group G_i which user u directly belonged to, check_weak(G_i),$u \in G_i$

```
if exist( [Gi,o,op]=1) then
   return true; exit check_auth;
elseif exist([Gi,o,¬op]=1) then
   return false; exit check_auth;
else go to (5);
```

(5) In order check the above control group G_j which user u indirectly belonged to, $u \in {}_1G_i \circ\circ\circ \in {}_mG_j \circ\circ\circ \in {}_nG_k$, (m<j<n):

```
while (exist(Gj, Gi∈Gj))
   redo check_weak(Gi);-- step (4)
   exit check_weak;
exit check_auth;  -- end of authorization
check
```

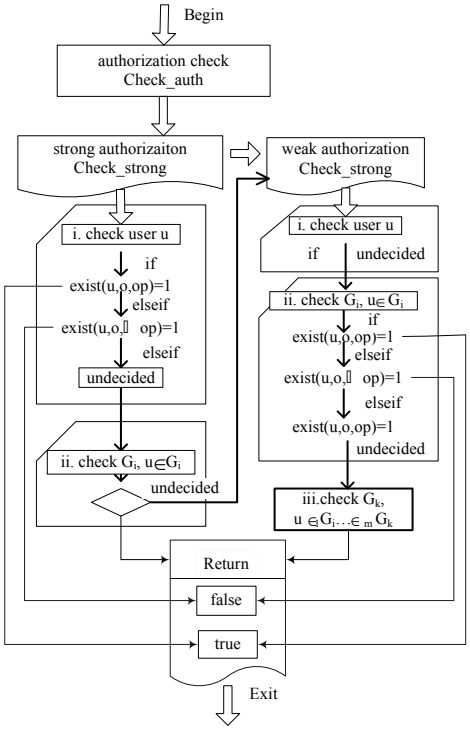

Fig. 5 Authorization check process

B. Implicit Authorization

Implicit authorization utilize**s** the existing authorization and authorization rules to deduce the new authorization. New authorization is contained in the existing authorizations and authorization rules [2, 14]. Implicit authorization improve the authorization efficiency to a certain extent, also recede the complexity of authorization model.

According to the derivative direction, implicit authorization can be divided into two kinds: Upset implicit authorization and

Downset implicit authorization. Implicit authorization means the inheritance of authorization, following will be the detailed illustrations on three facts: Subject, Object and access operation [8, 15]. Symbols "< " and " > " in the context represent the level relationships on the Object and access operation hierarchy.

On the view of subject, implicit authorization is closely connected to the Subject hierarchy.

(1) Upset implicit authorization on the subject hierarchy: If there exist the explicit authorization *(G_i, O, Op)* on the control group G_i, then will have implicit authorization *(G_k, O, Op)*, $G_i \in_n G_k$; If there exist the explicit negative authorization *(G_i ,O, ¬Op)* on the control group G_i, then to the control group G_j will have implicit authorization *(G_j, O, Op)*, $G_j \in_m G_i$

(2) Downset implicit authorization on the subject hierarchy: Control group on the subject hierarchy can realize the implicit on whole members in the group. If there exist the explicit authorization (G_k, O, Op) on the control group G_k, then to u_i, G_i where $u_i \in G_k$ and $G_i \in_i G_k$, there will be hold the implicit authorization (u_i, O, Op) and (G_i, O, Op).

If there exist the explicit positive authorization*(S, O, Op)* on *Op*, then to the *Op_* where *Op_*<*Op*(*Op_* represents the access operation on the lower level of *Op*) ,there will be *(S, O, Op_)*; If there exist the explicit negative authorization (*S, O, ¬Op*) on *Op*, then to the Op^- where Op^->*Op* (Op^- represents the access operation on the higher level of *Op*) ,there will be (*S, O, ¬* Op^-). Consequently, it is downset implicit authorization on the access operation hierarchy.

Certain object authorization on the Object hierarchy can be transferred to the lower object, and also can affect the higher object. Due to the flexibility and particularity of authorization object itself, the direction of implicit authorization is connected to the specific access operation.

Upset implicit authorization on the object hierarchy: If there exist the explicit positive authorization (*S ,O, Op*) of the subject *S* on the object *O*, then to the Object O^- where O^- is on the higher lever in the object hierarchy, there will be (*S,* O^-,*Op*); If there exist the explicit negative authorization (*S, O,¬Op*) of the subject S on the object O, then to the Object *O_* where *O_* is on the lower lever in the object hierarchy, there will be (*S, O_,Op*). For example, assume there exists *(s, obj, write)*, and object *obj* is an instance of class C, then there will be (*s, C, read*); Moreover, we can deduce the authorization (*s, obj, ¬read*) from (*s, C, ¬read*).

TABLE.I
IMPLICIT AUTHORIZATION TRANSFERRING ON THE SUBJECT, OBJECT AND ACCESS OPERATION HIERARCHY

	Explicit authorization	Upset implicit authorization	Downset implicit authorization
S	(G_i,o,op)	(G_k,o,op), $G_i \in_n G_k$	(u_i, o, op) , $u_i \in G_i$
	(G_i,o,¬op)	(G_j,o,¬op), $G_j \in_m G_i$	(G_j, o, op),$G_j \in_m G_i$
O	(s,o,op)	(s, o^-,op), o^->o	(s,o_,op), o_< o
	(s,o,¬op)	(s, o_,¬op), o_<o	(s,o^-,¬op), o^->o
Op	(s,o,op)	--	(s,o,op_), op_< op
	(s,o,¬op)	--	(s,o,¬op^-), op^->op

Downset implicit authorization on the object hierarchy: If there exist the explicit positive authorization *(S, O, Op)* of the subject *S* on the object *O*, then to the Object *O_* where *O_* is on the lower lever in the object hierarchy, there will be *(S, O_,Op)*; If there exist the explicit negative authorization*(S, O,¬Op)* of the subject S on the object *O*, then to the Object O^- where O^- is on the higher lever in the object hierarchy, there will be *(S,* O^-*, ¬Op)*. For example, assume there exists *(s, C, read)*, and *c* is an sub-class of class C, then there will be *(s, c, read)*; assume there exists *(s, obj, write)* and a_i is an attribute of object *obj*, then there will be *(s,* a_i*, write)* and *(s,* m_i*, execute)*; Moreover, assume *mic_obj* is a part-of the complex object *obj,* then there will be *(s, mic_obj, write)*.

C. Conflict and Overriding

On account to the Upset and Downset authorization transferring on each hierarchy, there might be conflict for one subject on the same object. Based on the Object hierarchy to control the conflict, commonly there exist following priority rules:

(1) Same subject on the same object for the same kind of access operation, weak authorization will be overridden by strong authorization and implicit will be overlapped by explicit authorization.

(2) For the conflict between two weak authorizations, the authorization for the members of control group Gi is always in preference to the control group *Gi*.

(3) The authorization inherited from the direct belonged control group is always in preference to the authorization inherited from indirect belonged control group.

(4) For the conflict between two strong authorization, the authorization occur later will be treated as invalid.

In brief, implicit authorization in a certain degree simplifies the authorization mechanism of object-oriented database. Implicit authorization accomplishes the authorization transfer on three facts subject, object and access operation. However, it to some extent lack of flexibility could not sufficiently satisfy the need of practice application. Since weak authorization holds the features of being overriding, it can be used in the exception disposal. Weak authorization can work together with negative authorization to localize the authorization invalidate level and node to accomplish the flexibly control of exceptions.

V. INSTANCE VALIDATE

Based on the above researches and summaries, this section tries to accomplish implement and validate. Shown as Fig. 6, define class Student and its subclass *grad_stud*, implement two instances of subclass: *grad_stud1* and *grad_stud2*.

On account to the subject hierarchy defined in Fig. 2 to carry on the validation of authorization mechanism:

(1) Assuming exists the explicit strong authorization (*G1, grad_student, update*) on control group *G1*:

```
GRANT update ON grad_student TO G1;
```

Since *U1,* ∈ *G1*, *U3* can get the implicit strong authorization *(U3, grad_student, update)* transferred from *G1* to hold the

read/write permission on the attributes, member methods and instance objects of subclass *grad_student*. Moreover, since *grad_student* inherited from *Student* class, *U3* can also get the read permission on the father class attributes: id and name.

If grant explicit negative authorization *(U3, grad_student, ¬read)* on *U3*:

```
NONGRANT read ON grad_student TO U3;
```

According to the priority principle of group members, U_3 will lose the read permission on class *grad_studen*t.

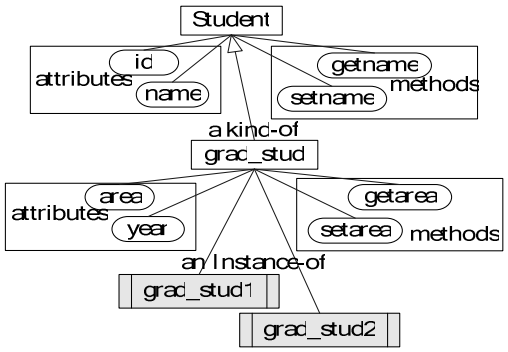

Fig. 6 Class hierarchy of instance

(2) Same preconditions as (1), and assuming exists the explicit positive authorization *(G_1, grad_student, update)* on G_1 and explicit positive authorization *(G_k, grad_student, ¬update)* on G_k. Since $U_1 \in G_1$, and $U_1 \in {}_1G_1 \in {}_2G_k$, according to the priority principle of direct belonged control group, U_1 can get the implicit authorization *(U_1, grad_student, update)*.

(3) Once U_1 gets the update permission on subclass *grad_student*, it will also get the read/write permission on all instance objects of *grad_studen*.

In the practical situation, generally one class have several lot of instances, the authorization to all instances of a class will be much too rough and general. Assume that in a particular situation, we don not want to grant U_3 read/write permission on certain instance (such as *grad_stud1*), but reserve the permission on remaining all other instances (such as *grad_stud2*). Then it will be needed to utilize weak authorization [*U_1, grad_student, update*] on U_1 to *grad_student*, and utilize negative read/write authorization (*U_1, grad_stud2, update*) on *grad_stud2 to* satisfy this flexible requirement.

```
WEAKLY GRANT update ON grad_student TO U1;
NONGRANT update ON grad_stud2 TO U1;
```

Oracle object system obeys to the basic features of object-oriented ideology. Define the instance structure in Oracle object system, shown as Fig. 6. Utilize the role to control the permission of control group and its members in Fig. 2. Use the grant and revoke command to implement the negative and positive authorization. Following results can be shown through the validate processes:

In brief, implicit authorization mechanism based on each hierarchy insecure mode to implement the permission transferring and conflict resolving. Moreover, combine the weak authorization with negative authorization to adjust the subject implicit authorization on particular object hierarchy can improve the flexibility and usability of authorization mechanism [3].

TABLE.II
INSTANCE VALIDATE RESULTS

S	Student.attr	grad_student	grad_stud1	grad_stud2
G_k	--	(¬update)	*¬update*	*¬update*
G_1	*read*	(update)	*update*	*update*
U_3	--	(¬read)	--	--
U_1	*read*	[update]	*update*	(¬update)

VI. CONCLUSION

OODB introduces the concept of Object-Oriented, which make the security protection mechanism of OODB more complex. Authorization provides suitable precision to access the specified users, and guaranteed to protect the securities of OODB in company with other security mechanisms. Implicit Authorization is based on the subject's hierarchies, objects hierarchies and access hierarchies of the security authorization modes, and simplifies the authorization mode, in addition, to combine with other authorization mechanisms, implicit authorization can make protection on the authorization of OODB expediently and effectively. Implicit Authorization introduces the authorization conflicts when the same subject accesses the object while it simplifies the complexity of the authorization mode. We also have a long way to go to make a more deep research and conclusion of the authorization conflicts.

REFERENCES

[1] Oki Y, Chikaraishi T, Shimomura T and Ohta T. A design method for data integrity in object-oriented database systems. International Conference on Information Engineering, Proceedings of IEEE Singapore, 1995: 204-209.

[2] Ambhore, Premchand B, Meshram B, and Waghmare V B. An implementation of object-oriented database security. Software Engineering Research, Management & Applications, 2007. SERA 2007. 5th ACIS International Conference, 2007:359-365.

[3] Fernandez E B, Gudes E, and Hauyan Song. A model for evaluation and administration of security in object-oriented database systems. Knowledge and Data Engineering, IEEE Transactions on Vol. 6, Issue 2, April 1994: 275-292.

[4] Bertino E, Bettini C, Ferrari E and Samarati P. A temporal access control mechanism for database systems [J]. Knowledge and Data Engineering, IEEE Transactions on Vol. 8, Issue 1, February. 1996: 67-80.

[5] Demurjian SA, Hu M Y, Ting T C and Kleinman D. Towards an Authorizaiton Mechanism for User-Role Based Security in an Object-Oriented Design model[C]. Computer and Communications, 1993, 12th Annual International Phoenix Conference on, March 1993:195-202.

[6] Thomas R K, Sandhu R S. A Trusted Subject Architecture for Multilevel Secure Object-oriented Database [J]. Knowledge and Data Engineering, IEEE Transactions on Vol. 8, Issue 1, Feb. 1996: 16-31.

[7] Bertino E, Jajodia S and Samarati P. Supporting multiple access policies in database systems[C]. Security and Privacy, 1996. Proceedings, 1996 IEEE Symposium on May1996: 94-107.

[8] Xu Jiepan, Object Oriented Database and Applications. Beijing: Science Press, 2003:78-95.

[9] Ni Xianjun. A logic specification and implementation approach for object-oriented database security. Knowledge Discovery and Data Mining, 2008. WKDD 2008. International Workshop, 2008: 461-464.

[10] Zhou Deyu, Luobin, Chen Shifu, Security Model for Object-Oriented Database Systems and its Application. Computer Engineering and Applications. 2003.27:210-212.

[11] Wang Yijie, Object Oriented Database. Beijing: Electronic Industry Press, 2003:253-267.
[12] Zhang Min, Xu Zhen, Feng Dengguo, Database Security. Beijing: Science Press, 2005: 140-146.
[13] Chen Qiang. The Problem About the Safe Protection of Object-oriented Database. Computer Engineering Vol. 24, No.6, 1994:41-43.
[14] Milen J K, Lunt T F. Security for object-oriented database systems. Research in Security and Privacy, 1992. Proceedings, 1992 IEEE Computer Society Symposium on May. 1992: 260-272.
[15] Zhang C N, Honglan Zhong. An Integrated approach for database security and fault tolerance. Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on Vol. 1, 2004: 762-766.