I.J. Engineering and Manufacturing **2011, 2, 28-32**
Published Online April 2011 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijem.2011.02.05

Available online at http://www.mecs-press.net/ijem

The Application of VPD in University Entrance Examination Management System

Wang Guan[a,*], Yan Rui[a]

[a] *Computer Institute Beijing University of Technology Beijing, China*

Abstract

Virtual Private Database (VPD) ensures the privacy of information and prevents direct data leakage through computer terminals by providing Fine-grained Access Control (FGAC). This paper firstly introduces the security requirements of the test database management system and based on which, further analyzes the existing shortcomings in access control technology. Through a brief introduction of VPD, the security design of the test database management system is then described, in which both the security and flexibility of the system are increased and the information leaks and other security related problems of the system are solved.

Index Terms: grained access control; database; virtual private database; security policy; row-level security

1. Introduction

The test database management system stores privacy sensitive data [1], which consists of large amount of test database questions and other tests. Any unauthorized view or modification will make great impact and cause huge loss. This system adopts C/S design pattern, the key data's [2] leakage is mainly through the computer terminals directly. In order to ensure the integrity, confidentiality and availability of the data in the database, it is not enough to only set security policies in the applications for data access control [3]. Malignant users may use some tools provided by database, such as SQL*PLUS to bypass the security mechanism of the application thus visit the database directly, which poses safety threat to data. This problem can only be solved by strengthening the database Fine-grained Access Control (FGAC) [4] to realize strict hierarchy of data access.

FGAC is an extension of Discretionary Access Control, which can take advantage of rules or policies to achieve fine access control with security policy. Granularity of access control can be set in accordance with specific requirements, so FGAC has higher requirements for flexibility and security. The implementation of

* Corresponding author.
E-mail address: wangguan@bjut.edu.cn

FGAC can effectively strengthen Discretionary Access Control and Mandatory Access Control.

2. The security requirements of test database management system

The test database management system stores a large amount of test questions. This system can manage the test information through different operations by different users to improve the test database quality and management level. The users of this system are various subject secretary and relevant leaders. Subject secretary can entry, modify, delete and view test questions in this subject. Relevant leaders can view all information of test questions.

The data in this system may suffer two security threats [5] in the process of handle and storage: (1) the integrity of information in the database may be destroyed, and (2) unauthorized access may cause the information be leaked and tampered.

Due to this system stores a large number of sensitive data, serious impact and losses will be caused if the system is damaged or the information is intercepted or modified. In order to ensure the security of information in the system, the system requires severely restricted functions and range of data which the users can visit through authority mechanism. Subject secretary can only examine the test questions in this subject; they cannot examine the information in other subject. For relevant leaders' visit, it can display all information of the test questions.

3. The shortcomings of existing access control

3.1. Setting Security Policies at the Application Layer

By embedding access control into the application, malignant users can use tools such as SQL*PLUS to bypass the security policies in application layer to visit the database directly. There are four shortcomings of this method: (1) change in access control policies need to change a lot of application codes, (2) security policies which are set at the application layer may be bypassed, (3) it is very difficult to implement access control at the application layer to ensure the codes security, programmers can easily set the trapdoor, and (4) it needs to implement security policies for each application, which will result in waste of resources.

Setting security policies at the application layer can easily be bypassed, which poses threat to the information. Therefore the FGAC technology should be adopted to set the security policies on data in the database.

3.2. To Use Authority Mechanism Which is Provided by Database

Oracle database [6] can provide the capacity to authorize users to access the object. But its access authority is defined at the object level. It can only be limited to the entire table instead of specific rows in the table. This approach is sufficient for many applications, but this system requires users can only visit the information in this subject. Therefore, more independent control for access and authorization is needed.

3.3. Using Views

Oracle can create views for different users and restrict users' access to the data table by authorizing the views to users to realize row-level security control of the datasheet. However, if users need to insert, update or delete some information, it must authorize the base table directly which will undermine the principle of independent operation of users. This method also needs to manage and maintain a lot of views which will bring a heavy management burdens to the database administrator. In some cases views can also be bypassed.

Using Virtual Private Database (VPD) [7] which provided by oracle dose not only realize row-level security control of the table, but also easy to manage. It only need to modify the corresponding policy functions, without having to modify the application codes.

4. Introduction of VPD

VPD, its row-level security feature provides a server-side enhanced and fine data access control. Row-level security is not open an entire table to users which have any access authority, but limit access to specific rows in the table.

VPD row-level security design only needs to build security policies in server for one time rather than implement security policies for each application. This will not only greatly save cost, but also eliminate the application-level security issues. Security policies are placed in the database, so that different applications and users can not bypass the security policies. It will eventually be automated access control by server-side policy function, no matter what application is used. Each user can only access the data he is authorized to visit, as everyone has a separate database.

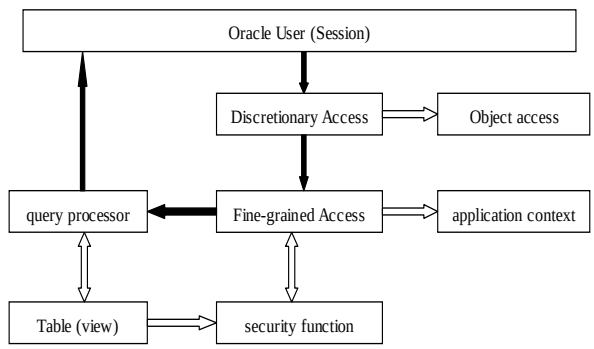

Fig. 1. VPD architecture

VPD architecture is shown in Fig. 1. The database system will call a stored procedure to set the value of pre-defined context when a user logs on the database. After that, if a user sent data manipulation statements to the table or the view which is associated with the policies, the database system firstly checks the user's discretionary access. If this access does not match, the program will exit and return an error message. Otherwise the system will be transferred to FGAC to process. Firstly, FGAC will determine what policies need to be implemented based on the type of statements. Policies can be embodied by the storage function which is able to return the predicate. This predicate will be used to dynamically modify the statement submitted by the user. Query processor executes the SQL statement which has been modified, and returns the final results to the user. This whole process is completely transparent to the user.

5. The security design of test database management system

There are two tables in this system: subuser table, which stores the users' information; subject table, which stores the test questions. The subuser table contains following fields: subuserid, subusername, subuserflag and so on. The users are mainly various subject secretaries and relevant leaders. The subject table contains some fields such as: subjectid, subuserid and subjectname.

In order to protect the information in the subject table from unauthorized view or modification, the system requires that the user can only access or operate partial contents of this table after he logged on the database according to the appropriate authority. So the security of data is ensured.

Adopting VPD makes each user see completely different data sets. When a subject secretary uses subusername to log on the system in the client, a one-to-many relationship is established between the legitimate user and the subject table by subuserid as the primary key. Test questions of their own subject can be found in this table.

According to the system security requirements, a security policy target is: various subject secretary can view the test questions in this subject; relevant leaders can view all information of the test questions.

5.1. Creation of the Application Context and Set the Package

Because a one-to-many relationship is established between the users of database and application in this system. It needs to create a user-defined application context and set the context attributes by creating the package and functions.

1) Creating a context which is named SUB_CTX: the attributes of this context are set by the pakeage of USER_SDE. The codes are:

```
create context SUB_CTX using SUB.USER_SDE.
```

2) Creating the package and function which set the context: the program can send the value of subusername which comes from the application and find the value of subuserid in subuser table. Then it will call dbms_session.set_context to set the attributes and the value of the context. The codes are:

```
create or replace package body USER_SDE is
procedure Get_Subuserid (ename in varchar2) is
v_subuser_id number;
begin
select subuserid into v_subuser_id from subuser where subusername=ename;
dbms_session. set_context (′SUB_CTX′,  ′SUBUSER_ID′, v_subuser_id);
end Get_Subuserid;
```

5.2. Creating a Security Policy Function

Once the security policy was associated with the table or the view, the policy function will return a value which is a where clause in the form of an access control condition when a user submits a query. This clause is attached to the SQL statements on the basis of the types. It is used to limit the rows which will be returned, update or delete.

Subuserid in the subject table can ensure the uniqueness of the test questions. The policy function should return a where clause which contains a value of subuserid in order to realize the security policy target. This function can find subuserid from application context and return a query clause which is added to the subject table. The codes are:

```
create or replace package body SUB_SDE as
function subjectid_sde (user_name in varchar2, obj_name in varchar2) return varchar2 is
sql_pred varchar2 (500);
begin
sql_pred:=’ ’;
if(sys_ context（"USERENV"，
'SESSION_USERFLAG'）='1') then
sql_pred:=null;
else
```

```
sql_pred:= 'subuserid =sys_ context
("SUB_CTX","SUBUSER_ID")';
return sql_pred;
end subjectid_sde;
```

5.3. Making the Policy Function Associated with the Table or the View

Oracle uses the package which is named DBMS_RLS to manage the security policies. There are policy addition, deletion, update and start or stop in this package.

A policy named sub_policy is added to the program. When it runs a select statement for the subject table in sub pattern, the policy will run SUB_SDE.subjectid_sde function. After it executes that command, any queries for the test questions will only return their own information. If there is a query statement like: select * from subject, it will automatically modify the statement to realize FGAC, because of the security policy function. The codes are:

```
EXEC
DBMS_RLS.ADD_POLICY('sub', 'subject', 'sub_policy', 'sub', 'SUB_SDE.subjectid_sde', 'SELECT');
```

This system uses VPD to achieve FGAC: (1) security of the system is increased and the problems of setting the security policies in the application are eliminated, (2) it is easy to implement and manage. An appropriate security policy implementation rather than code modification is required, and (3) row-level security design of VPD requires time to build security policy on server-side, so it reduces the cost.

6. summary

The test database management system uses VPD technology. VPD provides powerful row-level security feature and makes the security policies attached to the data instead of the application. It makes the implementation independent of the application system and ensures any operation can not bypass the security mechanisms of the system. It dose not only improve the security of the system and ensure the security of the information, but also increase the flexibility of the application system design. It reduces the complexity of application development and the burdens on database administrator.

References

[1] Jingmin He, Min Wang. Cryptography and Relational Database Management System, Interactive Dialogue with Educator from Across State, 2001, pp.273–284. (in Chinese)

[2] R.Agrawal and J.Kiernan, Watermarking Relational Databases, In the proceedings of the International Conference on Very Lgare Databases, 2002, pp.155–166.

[3] Cheng-rong Wu and Shi-yong Zhang, Database Access Control Models, Computer Engineering and Applications, 2002, pp.169–185. (in Chinese)

[4] Rizvi R, Mendelzon A and Sudarshan S, Extending Query Rewriting Techniques for Fine-Grained Access Control, 2004, pp.13–18.

[5] U Maheshwari, R Vingralek amd W Shapiro, How to build a Trusted Database System on Untrusted Storage. In the proceedings of Operating System Design and Implementation, 2000, pp.135–150.

[6] Denning D. E. Field Encryption and authentication. Chaum Ded.Advancds in Cryptology. Proceedings of Crypto's 83.Unvi.Of California.Santa Barbare.CA.1983. Plenum Press. 1984, pp.231–247.

[7] The Virtual Private Database in Oracle9i. An Oracle Technical White Paper.http://otn.oracle.com/deploy/security/oracle9ir2/pdf/vpd9ir2twp.pdf.