


***Fine-Grained Access
Control with Oracle8i's
Virtual Private Database
Feature***



IOUG-A Live 2001

Douglas Scherer

Core Paradigm



Agenda

- Overview of VPD
- Demonstration of VPD Usage
- Walkthrough of Implementation Process
 - Six Steps to Implementation
 - Review of Code Samples



What is the VPD?

- Fine-grained access control
 - Functionality that provides the ability to define access policies for individuals or groups of users to data resources
- Application context
 - The ability to securely populate and retrieve cached attributes containing information about your application environment



Uses for VPD

- Web and Client/Server Applications (primarily with different logons)
- Most places you would consider using views for access control. For example, allowing:
 - hospital physicians to see only the records of their own patients
 - customers to see only their own orders
 - managers to see only the employees in their own department



FAQs

- How does VPD work?
 - VPD works by dynamically adding a predicate to the DML or SELECT statement
- What's the difference: Roles vs. VPD?
 - Roles specify access to objects
 - VPD specifies access to the data set affected in/from those objects
- What's the difference: Views vs. VPD?
 - VPD allows different predicates depending on the SQL statement type
 - VPD introduces cached attributes functionality

A Look at the Implemented Rules

- For these examples
 - Add a column - ORACLE_USERNAME VARCHAR2(30) to scott.emp
 - As Scott, create a view named employees

```
CREATE OR REPLACE VIEW employees
AS SELECT *
FROM emp;
```

Demonstration
SCOTT, KING, SYSTEM

VPD Flow

Phase 1

1.1 User Logs-on;
DB Event Trigger
Fires

Application
Security
Package

1.2 Context
attributes are
populated

Context
Attributes

Phase 2

2.1 User Issues
SQL Statement
Against Object

SQL
Statement

Table/View
Security
Package

2.2 Add Predicate
to SQL Statement
if Policy is Defined



Six Steps to Implementing VPD

- 1) Plan the policy
- 2) Create the Context
- 3) Create an application security package that defines the attributes for the context
- 4) Set up automatic population of the application security attributes at logon
- 5) Create a table/view security package
- 6) Assign the policy to the table or view

1.a) List of Policy Access Rules ⇒



- Only the owner of the security package and Scott will have the right to issue SELECT or UPDATE statements directly against emp. All other's rights to directly SELECT from or UPDATE the emp table will be done via a view called employees



⇒ 1.a) List of Policy Access

Rules

- SELECT and UPDATE rights will be determined at runtime per logged-on employee:
 - Managers and above can SELECT all of employee records, but can update only the records of employees that they directly manage
 - Other employees can SELECT only the records of other employees in their department, and may not UPDATE any employee records
 - Anyone else logged on to the system (i.e. those who are not listed as employees in the emp table), cannot see any employees.



2.a) Create the Context

- Creating a context defines a context namespace and associates it with an application/procedure
- That procedure will be the only one allowed to populate that context's attributes



2.b) Create the Context in a Separate Security Schema

- Use a separate account to hold the application and table security packages
- For these examples use the schema HR_SECURITY_MANAGER
- Grant CREATE ANY CONTEXT privilege to HR_SECURITY_MANAGER
- Create the context

```
CREATE OR REPLACE CONTEXT hr_context  
USING hr_application_security;
```



3.a) Create an Application Security Package

- Defines the attributes for the context
- The application context attributes can be populated only from within the designated security package procedure specified in the CREATE CONTEXT statement
- A call to DBMS_SESSION.SET_CONTEXT outside of the application security package yields the following error: ORA-01031: insufficient privileges



3.b) DBMS_SESSION.SET_CONTEXT

- DBMS_SESSION.SET_CONTEXT is used to define and populate a context's attributes
- Takes Three Parameters
 - Namespace: the context namespace to use for the application context. The context must have previously been created via the CREATE CONTEXT command
 - Attribute: name of the attribute to be set
 - Value: the value that will be retrieved when accessing the attribute from SYS_CONTEXT (limited to 256 bytes)



3.c) SYS_CONTEXT

- SYS_CONTEXT is a function new in Oracle8i that returns the value of an attribute in a context namespace
- Takes Two Parameters
 - Context namespace: name of the context
 - Attribute name: name of the attribute in the context for which you want to get a value



3.d) SYS_CONTEXT: Example

- Example 1

```
SELECT SYS_CONTEXT ( 'USERENV' , 'IP_ADDRESS' )
      FROM dual;
IP_ADDRESS
-----
172.334.22.41
```

- Example 2

```
BEGIN
  DBMS_OUTPUT.PUT_LINE
    (SYS_CONTEXT ( 'HR_CONTEXT' , 'EMPNO' ) ) ;
END;
```

3.e) An Application Security Package

- There are no required parameters for the application security procedure.
- Create Application Security package as HR_SECURITY_MANAGER

Code Review

4.a) Set up Automatic Population of the Application Security Attributes

- Create a database event trigger that fires for each user connecting to the database at logon.
 - Use a non-SYS account with DBA privileges

```
CREATE OR REPLACE TRIGGER after_logon_al
  AFTER LOGON ON DATABASE
BEGIN
  hr_security_manager.hr_application_security.set_hr_
  context_attributes;
END; -- after_logon_al
```



4.b) Issues Regarding DB Event Triggers

- DB event triggers affect the database at a fundamental level
- If you have problems connecting after creating (or attempting to create) a DB event trigger, connect as – SYS AS SYSDBA - and drop or disable the trigger.



5.a) Create a Table/View Security Package

- The security function in the package will make use of the attributes in the application context to choose a predicate for the UPDATE or SELECT statement the user issues against employees.
- Create Table policy package as HR_SECURITY_MANAGER

Code Review

6.a) Assign the policy to the table or view

```
BEGIN
  DBMS_RLS.ADD_POLICY('SCOTT', 'EMPLOYEES',
    'SELECT_EMPLOYEES_POLICY',
    'HR_SECURITY_MANAGER',
    'hr_table_security.' ||
      'select_employees_predicate',
    'SELECT'
  );
  DBMS_RLS.ADD_POLICY('SCOTT', 'EMPLOYEES',
    'UPDATE_EMPLOYEES_POLICY',
    'HR_SECURITY_MANAGER',
    'hr_table_security.' ||
      'update_employees_predicate',
    'UPDATE'
  );
END;
```



6.b) Issues Regarding DBMS_RLS

- Issuing any of the procedures in the DBMS_RLS package will commit any uncommitted transactions in the current session



Good Metalink Documents

- Doc ID: Note 21168.1
 - Subject: Cost Based Optimizer Predicate Trace

Author Contact

- Douglas Scherer
 - <http://www.coreparadigm.com>
 - dscherer@coreparadigm.com
- *Oracle8i Tips and Techniques*
Osborne McGraw-Hill, Oracle Press,
ISBN 0072121033
- Oracle DBA Interactive Workbook
and Video Course
 - Prentice Hall: ISBN 0130157422
and 0130321230

