

Absicherung einer bestehenden Applikation mit Oracle Virtual Private Database VPD

Von Ernst Leber

August 2008



Absicherung einer bestehenden Applikation mit Oracle Virtual Private Database VPD

Von

Ernst Leber

Dieser Artikel zeigt, wie Oracle VPD in Oracle eingesetzt werden kann, um Daten nur mit einer vorgegebenen Applikation und deren Businesslogik ändern zu können, andere Tools hingegen zu sperren.

Aufgabe

Die Benutzer melden sich von einem Windows Client mit einem Sammelaccount an der Datenbank an und können ohne weitere Authentifizierung die Applikation nutzen. Dieses Verhalten darf nicht geändert werden. Zusätzlich soll sichergestellt werden, dass die Benutzer nur diese Anwendung und die darin enthaltene Businesslogik für Änderungen von Daten in der Datenbank nutzen. Das Sperren bzw. Freigeben von Benutzern und anderen Tools für diese Daten muss ohne Neuansmeldung sofort wirksam sein.

Umsetzung

Diese Aufgabe wird mit Oracle Virtual Private Database (VPD) gelöst. Oracle VPD ist Bestandteil der Enterprise Edition und stellt Methoden und Werkzeuge zur Verfügung, mit denen die Where-Bedingung einer SQL-Abfrage erweitert werden kann, um dynamisch das Ergebnis einer Abfrage zu verändern.

Dieses dynamische Erweitern der Where-Bedingung wird genutzt, um den Zugriff auf die Daten in der Datenbank zu steuern. Für diese Steuerung werden ein Datenbank-Package, Tabellen und eine Policy erstellt. Diese zusätzlich erforderlichen Elemente werden im Folgenden beschrieben.

Für die Realisierung wird ein Benutzer `app_admin` erstellt. In diesem Schema werden die zusätzlich erforderlichen Tabellen und die Logik erstellt. Als zu schützende Daten werden in diesem Beispiel die Inhalte der Tabelle „DEPARTMENTS“ aus dem HR Schema genutzt.

Für das Sperren und Freigeben von Benutzern und Tools wird der Anmeldename des Betriebssystems genutzt. Dieser Anmeldename wird in einer Zuordnungstabelle Applikationen zugeordnet. Anwendungen oder Tools, die nicht in der Zuordnungstabelle eingetragen sind, haben keinen Zugriff auf die Daten in der Datenbank.

Es sind 3 zusätzliche Tabellen nötig:

- Benutzertabelle, `TAB_BENUTZ`, in dieser Tabelle werden die Windows Anmeldenamen der User eingetragen, dieser Name kann mit `sys_context('USERENV','OS_USER')` ausgelesen werden

- Applikationstabelle TAB_APPLIK, in dieser Tabelle werden die erlaubten Applikationen, z.B. der Name der Business Applikation eingetragen, der mit `sys_context('USERENV','MODULE')` ausgelesen werden kann
- Zuordnungstabelle ZUO_BENAPP, in dieser Tabelle werden die Benutzer und erlaubten Applikationen über die IDs zugeordnet

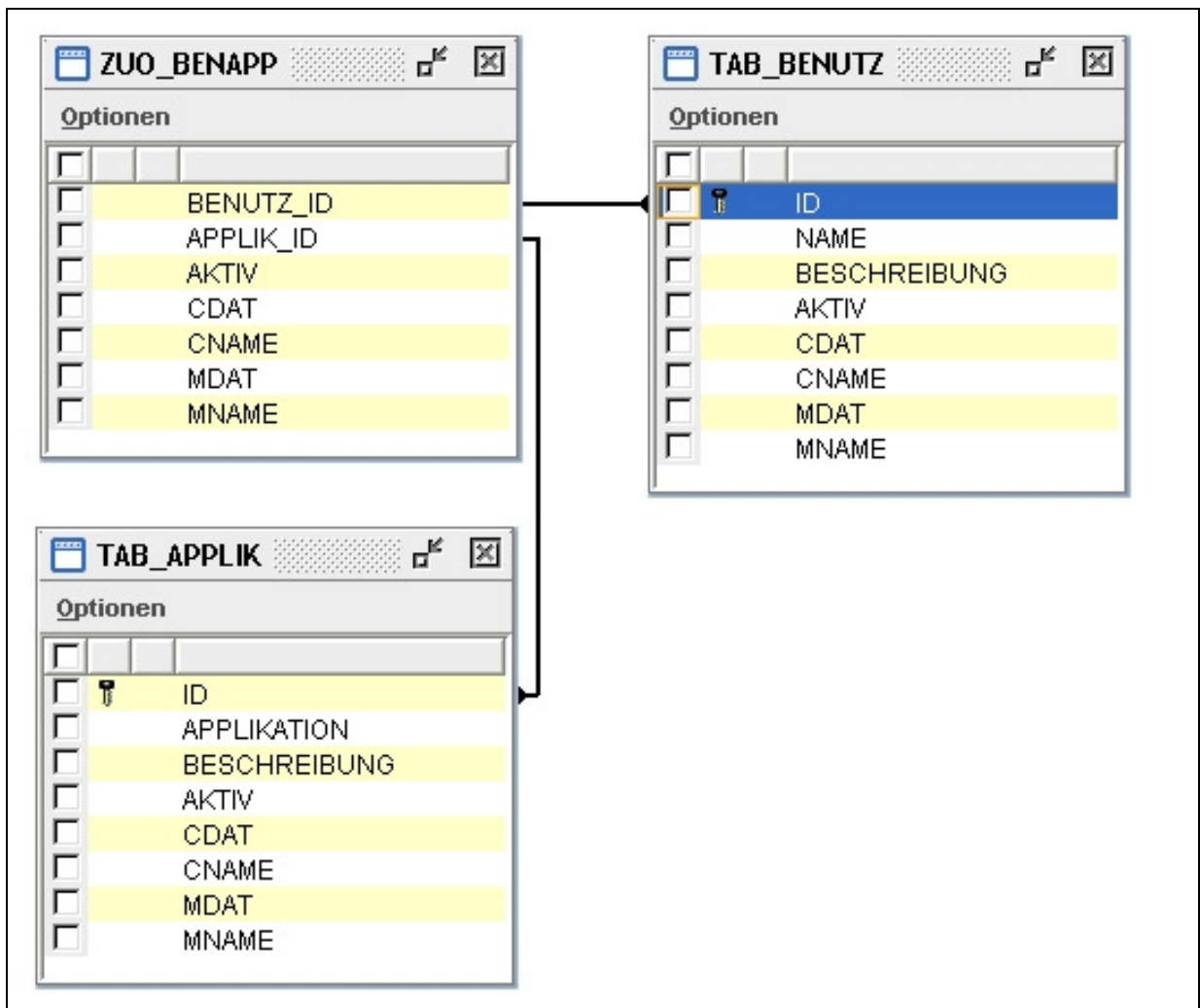


Abb. 1: Das ER Modell der erforderlichen Tabellen

Erstellungs- bzw. Änderungsdatum und Namen („CDAT“, „CNAME“, „MDAT“ und „MNAME“) werden durch Trigger automatisch in den Tabellen gespeichert.

Eine Applikation, die in der Tabelle TAB_APPLIK eingetragen ist, kann durch Eintragen von 0 in der Spalte „AKTIV“ mit sofortiger Wirkung gesperrt werden. Soll die Sperrung benutzerspezifisch erfolgen, geschieht dies entweder durch generelles Sperren des Benutzers in der Tabelle TAB_BENUTZ oder durch Sperren einer einzelnen Applikation für den Benutzer in der Zuordnungstabelle ZUO_BENAPP.

Im Package SEC_APPLIK werden diese Informationen zur Laufzeit in der Funktion set_Applik ausgewertet und der Zugriff auf die Daten der Beispieltabelle freigegeben bzw. gesperrt. Diese Funktion muß als Parameter den Schema-Namen und die Tabelle haben und liefert einen String zurück, um den die Where-Bedingung erweitert wird.

```
create or replace
package body sec_applik
as
  function set_applik(schema_name in varchar2,object_name in varchar2)
  return varchar2
  is
    l_where varchar2(1000):='';
    l_count  NUMBER;
  begin
    select count(*)
    into l_count
    from zuo_benapp zbp
    join tab_applik tak
      on tak.id = zbp.applik_id
    join tab_benutz tbz
      on tbz.id = zbp.benutz_id
    where tak.applikation = sys_context('USERENV','MODULE')
    and tak.aktiv = 1
    and ( instr(tbz.name,sys_context('USERENV','OS_USER')) > 0
        or instr(sys_context('USERENV','OS_USER'),tbz.name) > 0 )
    and tbz.aktiv = 1
    and zbp.aktiv = 1
    ;
    if l_count >= 1 then
      l_where := '';
    else
      l_where := '1=2';
    end if;
    return l_where;
  end;
end;
```

Abb. 2: Listing des Packages app_admin.sec_applik

Mit einer Policy wird die Funktion sec_applik.set_applik im System bekannt gemacht und mit der Tabelle „DEPARTMENTS“ aus dem HR Schema verknüpft. Diese Policy gilt für Select, Insert, Update und Delete Rechte.

```

begin
  dbms_rls.add_policy(
    object_schema => 'HR',
    object_name => 'DEPARTMENTS',
    policy_name => 'APP_BLOCK',
    function_schema => 'APP_ADMIN',
    policy_function => 'SEC_APPLIK.SET_APPLIK',
    statement_types => 'select, insert, update, delete',
    update_check => TRUE,
    enable => TRUE,
    static_policy => FALSE
  );
end;

```

Abb. 3 Listing der Policy

Test

Der einfachste Test für diese Funktion ist, zunächst nichts in die Steuertabellen TAB_BENUTZ, TAB_APPLIK und ZUO_BENAPP einzutragen.

```

SQL*Plus: Release 10.2.0.3.0 - Production on So Mai 18 10:44:26 2008
Copyright (c) 1982, 2006, Oracle. All Rights Reserved.

SQL> conn hr/hr
Connect durchgeführt.
SQL> select * from departments;

Es wurden keine Zeilen ausgewählt

SQL> desc departments;
Name                                Null?    Typ
-----
DEPARTMENT_ID                       NOT NULL NUMBER(4)
DEPARTMENT_NAME                      NOT NULL VARCHAR2(30)
MANAGER_ID                           NUMBER(6)
LOCATION_ID                            NUMBER(4)

SQL> insert into departments values(1,'test',12,3);
insert into departments values(1,'test',12,3)
*
FEHLER in Zeile 1:
ORA-28115: Policy mit Verletzung von Check-Option

SQL> delete from departments where department_id=20;

0 Zeilen wurden gelöscht.

SQL> rollback;

Transaktion mit ROLLBACK rückgängig gemacht.

SQL> |

```

Abb. 4: Tests im Schema HR ohne Freigabe

Nach dem Eintragen und Freigeben Ihres OS_USER Namens und des Applikationsnamens in die Tabellen TAB_BENUTZ, TAB_APPLIK und ZUO_BENAPP sind die Daten sichtbar und können geändert werden.

```
SQL*Plus: Release 10.2.0.3.0 - Production on So Mai 18 11:03:00 2008
Copyright (c) 1982, 2006, Oracle. All Rights Reserved.

SQL> conn app_admin/app_admin
Connect durchgeführt.
SQL> insert into tab_applik (applikation, beschreibung, aktiv)
  2  select sys_context('USERENV','MODULE'), 'SQL*Plus',1 from dual;

1 Zeile wurde erstellt.

SQL> insert into tab_benutz (name, beschreibung, aktiv)
  2  select sys_context('USERENV','OS_USER'), 'SQL*Plus',1 from dual;

1 Zeile wurde erstellt.

SQL> insert into zuo_benapp (benutz_id, applik_id, aktiv)
  2  SELECT tab_benutz.id, tab_applik.id, 1 from tab_benutz, tab_applik;

1 Zeile wurde erstellt.

SQL> commit;

Transaktion mit COMMIT abgeschlossen.

SQL> |
```

Abb 5: Eintragen und Freigeben der Applikation und des Users in die Steuertabellen

```

SQL*Plus: Release 10.2.0.3.0 - Production on So Mai 18 11:13:52 2008
Copyright (c) 1982, 2006, Oracle. All Rights Reserved.

SQL> conn hr/hr
Connect durchgeführt.
SQL> desc departments
Name                                     Null?    Typ
-----
DEPARTMENT_ID                           NOT NULL NUMBER(4)
DEPARTMENT_NAME                          NOT NULL VARCHAR2(30)
MANAGER_ID                               NUMBER(6)
LOCATION_ID                                NUMBER(4)

SQL> select * from departments where department_id = 10;
DEPARTMENT_ID DEPARTMENT_NAME          MANAGER_ID LOCATION_ID
-----
           10 Administration                200          1700

SQL> update departments set location_id = 1000 where department_id = 10;
1 Zeile wurde aktualisiert.

SQL> select * from departments where department_id = 10;
DEPARTMENT_ID DEPARTMENT_NAME          MANAGER_ID LOCATION_ID
-----
           10 Administration                200          1000

SQL> rollback;

Transaktion mit ROLLBACK rückgängig gemacht.

SQL> select * from departments where department_id = 10;
DEPARTMENT_ID DEPARTMENT_NAME          MANAGER_ID LOCATION_ID
-----
           10 Administration                200          1700

SQL> |

```

Abb. 6: Daten können mit SQL*Plus gelesen und geändert werden.

Fazit

In diesem Beispiel wurde VPD genutzt, um den Datenzugriff auf bestimmte Applikationen zu begrenzen. Weitere Tabellen werden einfach durch Erstellen analoger Policies geschützt. Auf diesem Weg ist im praktischen Einsatz z.B. der Zugriff mit TOAD, SQL Developer oder Excel und Microsoft Query via ODBC unterbunden worden.

Mit Secure Roles ist es ebenfalls möglich, den Datenzugriff zu kontrollieren, die erforderlichen Rollen müssen aber explizit gesetzt werden, was Änderungen in der Applikation bedeutet. Ebenso sind Lösungen über Views und Trigger denkbar, die aber einen erhöhten Overhead im System zur Folge haben. (Im Anhang befindet sich ein komplettes Listing mit allen SQL-Befehlen zum Erstellen und dieser Demo.)

Anhang Listing SQL-Befehle

Create User

```
connect system/manager
create user app_admin identified by app_admin default tablespace users temporary table-
space temp;
grant create session to app_admin;
grant create any context to app_admin;
grant create table to app_admin;
grant unlimited tablespace to app_admin;
grant create sequence to app_admin;
grant create procedure to app_admin;
grant create trigger to app_admin;
connect sys/change_on_install as sysdba
grant execute on dbms_ols to app_admin;
grant execute on dbms_session to app_admin;
```

Erstellen der Tabellen

```
connect app_admin/app_admin
CREATE TABLE TAB_APPLIK
( ID          NUMBER NOT NULL ENABLE,
  APPLIKATION VARCHAR2(50 BYTE),
  BESCHREIBUNG VARCHAR2(255 BYTE),
  AKTIV       NUMBER(1,0),
  CDAT        DATE,
  CNAME       VARCHAR2(50 BYTE),
  MDAT        DATE,
  MNAME       VARCHAR2(50 BYTE),
  CONSTRAINT APPLIK_PK PRIMARY KEY (ID));

CREATE SEQUENCE SEQ_APPLIK MINVALUE 1 INCREMENT BY 1 START WITH 1
ORDER NOCACHE NOCYCLE;
```

```
CREATE OR REPLACE TRIGGER TRG_BIU_APPLIK
BEFORE INSERT OR UPDATE ON TAB_APPLIK
FOR EACH ROW
BEGIN
  if inserting then
    select SEQ_APPLIK.nextval, sysdate, user into :new.id, :new.cdat, :new.cname from
dual;
  end if;
  if updating then
    select sysdate, sys_context('USERENV', 'OS_USER') into :new.mdat, :new.mname from
dual;
```

```
end if;  
END;
```

```
CREATE TABLE TAB_BENUTZ  
( ID NUMBER,  
  NAME VARCHAR2(50 BYTE),  
  BESCHREIBUNG VARCHAR2(255 BYTE),  
  AKTIV NUMBER(1,0),  
  CDAT DATE,  
  CNAME VARCHAR2(50 BYTE),  
  MDAT DATE,  
  MNAME VARCHAR2(50 BYTE),  
  CONSTRAINT BENUTZ_PK PRIMARY KEY (ID)  
);
```

```
CREATE SEQUENCE SEQ_BENUTZ MINVALUE 1 INCREMENT BY 1 START WITH 1  
ORDER NOCACHE NOCYCLE;
```

```
CREATE OR REPLACE TRIGGER TRG_BIU_BENUTZ  
BEFORE INSERT OR UPDATE ON TAB_benutz  
FOR EACH ROW  
BEGIN  
  if inserting then  
    select seq_benutz.nextval, sysdate, sys_context('USERENV', 'OS_USER') into :new.id,  
:new.cdat, :new.cname from dual;  
  end if;  
  if updating then  
    select sysdate, sys_context('USERENV', 'OS_USER') into :new.mdat, :new.mname from  
dual;  
  end if;  
END;
```

```
CREATE TABLE ZUO_BENAPP  
( BENUTZ_ID NUMBER NOT NULL,  
  APPLIK_ID NUMBER NOT NULL,  
  AKTIV NUMBER(1,0) NOT NULL,  
  CDAT DATE,  
  CNAME VARCHAR2(50 BYTE),  
  MDAT DATE,  
  MNAME VARCHAR2(50 BYTE),  
  CONSTRAINT BENUTZ_BENUTZ_FK FOREIGN KEY (BENUTZ_ID)  
REFERENCES TAB_BENUTZ (ID) ENABLE,  
  CONSTRAINT BENROL_APPLIK_FK FOREIGN KEY (APPLIK_ID)  
REFERENCES TAB_APPLIK (ID) ENABLE  
);
```

```
ALTER TABLE zuo_benapp ADD CONSTRAINT BENAPP_PK PRIMARY KEY (BENUTZ_ID, APPLIK_ID);
```

```
CREATE OR REPLACE TRIGGER TRG_BIU_BENAPP
  BEFORE INSERT OR UPDATE ON zuo_benapp
  FOR EACH ROW
  BEGIN
    if inserting then
      select sysdate, sys_context('USERENV', 'OS_USER') into :new.cdat, :new.cname from
      dual;
    end if;
    if updating then
      select sysdate, sys_context('USERENV', 'OS_USER') into :new.mdat, :new.mname from
      dual;
    end if;
  END;
```

Create Package

```
create or replace package sec_applik
```

```
as
```

```
function set_applik(schema_name in varchar2, object_name in varchar2)
```

```
return varchar2;
```

```
end;
```

```
create or replace package body sec_applik
```

```
as
```

```
function set_applik(schema_name in varchar2,object_name in varchar2)
```

```
return varchar2
```

```
is
```

```
l_where varchar2(1000):="";
```

```
l_count NUMBER;
```

```
begin
```

```
select count(*)
```

```
into l_count
```

```
from zuo_benapp zbp
```

```
join tab_applik tak
```

```
on tak.id = zbp.applik_id
```

```
join tab_benutz tbz
```

```
on tbz.id = zbp.benutz_id
```

```
where tak.applikation = sys_context('USERENV','MODULE')
```

```
and tak.aktiv = 1
```

```
and ( instr(tbz.name,sys_context('USERENV','OS_USER')) > 0
```

```
or instr(sys_context('USERENV','OS_USER'),tbz.name) > 0 )
```

```
and tbz.aktiv = 1
```

```
    and zbp.aktiv      = 1;
  if l_count >= 1 then
    l_where := '';
  else
    l_where := '1=2';
  end if;
  return l_where;
end;
end;
```

Listing der Policy

```
begin
  dbms_rls.add_policy(
    object_schema => 'HR',
    object_name   => 'DEPARTMENTS',
    policy_name   => 'APP_BLOCK',
    function_schema => 'APP_ADMIN',
    policy_function => 'SEC_APPLIK.SET_APPLIK',
    statement_types => 'select, insert, update, delete',
    update_check  => TRUE,
    enable        => TRUE,
    static_policy => FALSE
  );
end;
```

Eintragen der Daten in die Steuertabellen

```
insert into tab_applik (applikation, beschreibung, aktiv)
select sys_context('USERENV','MODULE'), 'SQL*Plus',1 from dual;
insert into tab_benutz (name, beschreibung, aktiv)
select sys_context('USERENV','OS_USER'), 'SQL*Plus',1 from dual;
insert into zuo_benapp (benutz_id, applik_id, aktiv)
SELECT tab_benutz.id, tab_applik.id, 1 from tab_benutz, tab_applik;
```

Der Autor



Ernst Leber ist Senior Berater bei der MT AG, Ratingen.

Er besitzt langjährige Erfahrung mit Oracle Datenbanken. Schwerpunkte sind Arbeiten mit verteilten Datenbanken, PL/SQL Programmierung, Forms und Security Themen.

Sie können ihn per E-Mail erreichen unter:

ernst.leber@mt-ag.com

© Copyright 2008, MT AG Ratingen

Balcke-Dürr-Allee 9

40882 Ratingen

Tel. +49 (0) 21 02 309 61-0

Fax +49 (0) 21 02 309 61-10

info@mt-ag.com

www.mt-ag.com