PeteFinnigan.com Limited

Oracle Security Expertise

UKOUG Unix SIG, January 22nd 2008

Using Oracle VPD In The Real World

By

Pete Finnigan

Written Friday, 27th December 2007

Introduction - Commercial Slide.☹

- PeteFinnigan.com Limited
- Founded February 2003
- CEO Pete Finnigan
- Clients UK, States, Europe
- Specialists in researching and securing Oracle databases
- http://www.petefinnigan.com
- Consultancy and training available
- Author of Oracle security step-by-step
- Published many papers, regular speaker (UK, USA, Slovenia, Norway, more)
- Member of the Oak Table

Agenda

- What is VPD, is it used, info?
- Differences in various Oracle versions
- Securing VPD – often not considered
- Attacking VPD
- Problems – performance – design
- Conclusions

What Is VPD

- Called Virtual Private Database (VPD)
- Called Row Level Security (RLS)
  - Also DBMS_RLS controls it
- Called Fine Grained Access Control (FGAC)
- VPD includes:
  - Fine Grained Access Control
  - Application Contexts
  - Global Application Contexts

Is VPD Used In Anger?

- In my experience not much – why?
  - I have worked with a few clients to implement VPD
  - It is free with EE; not a cost option that may put people off like OLS
- Oracle are increasingly using it
  - In XDB ACL's
  - In E-Business Suite
  - As part of Database Vault and Audit Vault

Where To Find Information

- Effective Oracle database 10g security by design - **ISBN-13:** 978-0072231304
- RLS chapter - http://www.devshed.com/c/a/Oracle/RowLevel-Security-with-Virtual-Private-Database/
- Does VPD, FGA or audit really cause performance issues - http://www.insight.co.uk/files/presentations/Does%20VPD,%20FGA%20or%20Audit%20Cause%20Performance%20Issues.pdf
- Oracle Row Level Security - http://www.securityfocus.com/infocus/1743
- Row Level Security - http://www.dbazine.com/oracle/or-articles/jlewis15

VPD Through The Versions

- Row Level Security added in 8.1.5 release
- 9i adds multiple policies per table and policy groups controlled by application driving context
- 9i adds global contexts for connection pooling
- 10g adds column level policies, column masking, policy types (5) added for performance to allow caching, contexts updated to allow values to be passed to parallel slaves.
- 11g provides integration for Enterprise manager for Row Level Security Policies.

Securing VPD

- Leaking predicates
- Leaking policies
- RBAC on VPD structure / configuration
- Bypassing VPD by means of exception
- SQL Injection issues
- Direct data access

Finding the Predicate

- There are a number of possibilities to find predicates and details
  - Event 10730
  - Event 10060
  - V$vpd_policy – no one has access by default
- Library cache dump? – if static data present can also be leaked
- SGA can be dumped for binds, SQL, optimizer and more
- Common denominator – ALTER SESSION / SYSTEM / trace (many options - http://www.petefinnigan.com/ramblings/how_to_set_trace.htm)

Create A Simple Policy

- See code http://www.petefinnigan.com/vpd.sql
- Create a user PXF,
  - do some grants,
  - Create a table (copy of scott.emp)
  - Create a predicate function to block “deptno != 10”
  - Create a policy on pxf.emp
  - Number of rows restricted by 3
  - Demo!

Example

```
who_has_priv: Release 1.0.3.0.0 - Production on Wed Jan 16 19:13:16 2008
Copyright (c) 2004 PeteFinnigan.com Limited. All rights reserved.

PRIVILEGE TO CHECK        [SELECT ANY TABLE]: ALTER SESSION
OUTPUT METHOD Screen/File                [S]: S
…
Privilege => ALTER SESSION has been granted to =>
======================================================================
        Role => DBA (ADM = YES) which is granted to =>
                User => SYS (ADM = YES)
                User => SYSMAN (ADM = NO)
                User => SYSTEM (ADM = YES)
                User => TESTUSER (ADM = NO)
        User => SYS (ADM = NO)
        User => IX (ADM = NO)
        User => SH (ADM = NO)
        Role => RECOVERY_CATALOG_OWNER (ADM = NO) which is granted to =>
                User => SYS (ADM = YES)
        User => BI (ADM = NO)
        User => CTXSYS (ADM = NO)
        Role => OLAP_USER (ADM = NO) which is granted to =>
                User => SYS (ADM = YES)
        User => SCOTT (ADM = NO)
        User => HR (ADM = NO)
        User => DMSYS (ADM = NO)
        User => XDB (ADM = NO)
```

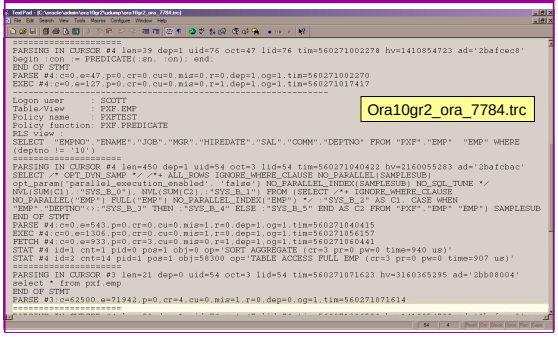

Example (2)

```
SQL> alter session set sql_trace=true;
Session altered.
SQL> alter session set events '10730 trace name context forever';
Session altered.
SQL> select * from pxf.emp;

     EMPNO ENAME      JOB              MGR
   COMM
---------- ---------- --------- ----------
   ---------
    DEPTNO
----------
      7369 SMITH      CLERK            7902 17-DEC-80        800
        20
…
SQL> alter session set events '10730 trace name context off';
Session altered.
SQL> alter session set sql_trace=false;
Session altered.
SQL>
```

As a normal user – SCOTT - I am able to determine the rules VPD imposes on me

Example (3)



Leaking Policy Details

- To secure VPD all of the configuration must be secured including:
  - %_POLICY_GROUPS
  - %_POLICY_CONTEXTS
  - %_POLICIES
- Access to functions must also be protected;
  - Definitions – OBJ$, SOURCE$, PROCEDURE$, ARGUMENT$

Example

```
who_can_access: Release 1.0.3.0.0 - Production on Wed Jan 16
   19:30:16 2008
Copyright (c) 2004 PeteFinnigan.com Limited. All rights reserved.

NAME OF OBJECT TO CHECK       [USER_OBJECTS]: ALL_POLICIES
OWNER OF THE OBJECT TO CHECK          [USER]: SYS
OUTPUT METHOD Screen/File                [S]: S
FILE NAME FOR OUTPUT              [priv.lst]:
OUTPUT DIRECTORY [DIRECTORY  or file (/tmp)]:
EXCLUDE CERTAIN USERS                    [N]:
USER TO SKIP                       [TEST%]:

Checking object => SYS.ALL_POLICIES
================================================================
   ===

Object type is => VIEW (TAB)
        Privilege => SELECT is granted to =>
        Role => PUBLIC (ADM = NO)
```

Madness by default anyone can see what policies exist that affect them

Example(2)

```
SQL> select object_owner,object_name,policy_name,
  2       pf_owner,pf_owner,function
  3  from all_policies;

OBJECT_OWNER                   OBJECT_NAME
------------------------------ ------------------------------
POLICY_NAME                    PF_OWNER
------------------------------ ------------------------------
PF_OWNER                       FUNCTION
------------------------------ ------------------------------
SCOTT                          EMP
PXFPOL                         SYS
SYS                            HACK

PXF                            EMP
PXFTEST                        PXF
PXF                            PREDICATE

2 rows selected
```

As SCOTT I could find out the predicate, I can also find out the policies that affect me.

RBAC on VPD structure

- RBAC must be applied on
  - Packages – DBMS_RLS, DBMS_SESSION
  - Policies – see previous slide
  - Policy functions, structure, source code
  - Contexts, application and global
  - Supporting data – static look up data
  - System privileges used
  - Grants on access to any of the above
- Don't just rely on VPD to protect data

Bypassing VPD

- VPD configuration should be designed normally to work with users (end users / identities)
  - i.e. access to groups of data is based on actual people, this is reflected in the VPD
- This is often done in total or part using application contexts – These are tied to the session
- BUT, they must use static data, session data, application data (i.e. FND_PROFILES) to ascertain who is who
- Whilst the context is reasonably secure often the data used could be changed/bypassed/spoofed
- All of the identity must be considered and hardened

Exempt Access Policy

```
who_has_priv: Release 1.0.3.0.0 - Production on Wed Jan 16 16:26:56
   2008
Copyright (c) 2004 PeteFinnigan.com Limited. All rights reserved.

PRIVILEGE TO CHECK        [SELECT ANY TABLE]: EXEMPT ACCESS POLICY
OUTPUT METHOD Screen/File                [S]: S
FILE NAME FOR OUTPUT              [priv.lst]:
OUTPUT DIRECTORY [DIRECTORY  or file (/tmp)]:
EXCLUDE CERTAIN USERS                    [N]:
USER TO SKIP                      [TEST%]:

Privilege => EXEMPT ACCESS POLICY has been granted to =>
======================================================================
        User => X (ADM = NO)

PL/SQL procedure successfully completed.

For updates please visit http://www.petefinnigan.com/tools.htm

SQL>
```

http://www.petefinnigan.com/who_has_priv.sql

SQL Injection

- SQL Injection could be used in a number of ways to exploit VPD:
  - Litchfield shows how to inject a call to DBMS_RLS.DROP_POLICY via XDB.XDB_PITRIG_PKG.PITRIG_DROP – see http://www.databasesecurity.com/dbsec/ohh-defeating-vpd.pdf
  - Many exploits from sites such as http://milw0rm.com can be used in the same way
  - Packages that expose VPD – see next slide
  - Applications that VPD could have components exploited – i.e. if the predicate is "constructed" using concatenation it could be exploited.

Ways To Access Policies

```
SQL> select owner,name,type
  2  from dba_dependencies
  3  where referenced_name='DBMS_RLS';

OWNER        NAME                           TYPE
------------ ------------------------------ ------------
PUBLIC       DBMS_RLS                       SYNONYM
SYS          DBMS_RLS                       PACKAGE BODY
SYS          LTUTIL                         PACKAGE BODY
SYS          LTADM                          PACKAGE BODY
XDB          DBMS_XDBZ0                     PACKAGE BODY
XDB          DBMS_XDBZ0                     PACKAGE BODY

SQL> select grantee,table_name from dba_tab_privs
  2  where table_name in ('LTUTIL','LTADM','DBMS_XDBZ0');

GRANTEE                        TABLE_NAME
------------------------------ ------------------------------
WMSYS                          LTADM
WMSYS                          LTUTIL
IMP_FULL_DATABASE              LTADM
PUBLIC                         DBMS_XDBZ0
```


Access The Data Directly

- Strings on data files
- With C or Java from the database
- Hex editors – Unix or Windows
- Block dumps – recent forensics papers cover
- Tools like bbed, CBAT, DUL like tools such as Ora*Dude and more
- Backups
- Exports
- Reports and lists of data from privileged users
- More?

Again do not consider VPD as a "be all" and and "end all" – work out where the data is and how it "flows"

Example (1)

```
SQL> select distinct dbms_rowid.rowid_block_number(rowid) blk,
  2     dbms_rowid.rowid_relative_fno(rowid) fno
  3  from pxf.emp;

       BLK        FNO
---------- ----------
       420          4

1 row selected.

SQL> select file_name from dba_data_files
  2  where file_id=4;

FILE_NAME
----------------------------------------
C:\ORACLE\ORADATA\ORA10GR2\USERS01.DBF

1 row selected.
```


Example (2)

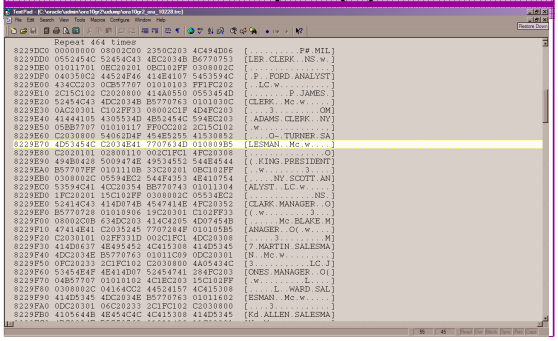
```
SQL> alter system dump datafile 4 block 420;
System altered.
SQL> connect sys/change_on_install as sysdba
Connected.
SQL> select * from pxf.emp where deptno=10;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL
   COMM
---------- ---------- --------- ---------- --------- ---------- -
   ---------
    DEPTNO
----------
      7782 CLARK      MANAGER         7839 09-JUN-81       2450
        10

      7839 KING       PRESIDENT            17-NOV-81       5000
        10

      7934 MILLER     CLERK           7782 23-JAN-82       1300
        10
```

Example (3)



Screens Can Break

- Certification and Support for Third party products - http://blogs.oracle.com/schan/newsItems/departments/extendingApps/2006/05/18#a200
- Adding VPD can break existing applications and other modules
- E-Business Suite screens have been seen to break because VPD is enabled
- There is often a fear with VPD implementers that they are not supported if VPD breaks something
- You can get into a complex support / certification saga
- If Oracle can reproduce – even if you let support have your code or an example with the same problem Oracle can help look at the issue

Layered Approach

- VPD must be part of a layered approach to securing data in an Oracle database
- RBAC on
  - Data
  - Security measures and policies
- Encryption for critical data
- Hardening must be done
- VPD as part of an overall solution
- Network security
- Audit trails
- More…

Performance

- VPD is often perceived as being bad due to perceived optimizer changes – aim to not excessively change the optimizer
- Often runs faster when VPD is enabled – less rows returned!
- Don't use excessive code in predicates i.e. select from dual or worse big tables
- Use indexes on the predicate columns
- Use static data if at all possible
- Use static policies if possible
- Keep the policy functions as simple as possible – good design is king!

Cached Policies and sys_context

- Another lesson learned was to pass back sys_context('…','…') rather than resolve the sys_context in the policy function
- 5 types of caching can be used:
  - Static – execute once, store predicate in SGA
  - Shared_static – cache predicate across multiple objects using same policy
  - Context_sensitive – use for connection pooling, server executes policy function on statement execution if a context change detected
  - Shared_context_sensitive – as above; shared across multiple objects; same policy
  - Dynamic – no caching executed every time

Design It First

- One of the key lessons I have learned with VPD is to design carefully first. Include:
- Business rules first (who/what/when)
- Identify the data to be protected
- Simplicity is the key – keep the rules / policies very simple (as simple as possible)
- Work out the identities, the rules for all access, the default state,
- Then design the contexts, predicates
- Test – create boundary tests as well

Multiple Policy Issues

- An example from the trenches
- A single table is needed as part of every predicate
- A lot of other tables access this table as part of the predicate generation
- A lot of policies created, identities designed, contexts created
- Problem: The single table cannot be protected with VPD as it breaks all other policies
- VPD needs, hardening, RBAC etc as well as a “complete” solution

Conclusions

- Looked at “what is VPD”
- What can it do
- How VPD can be bypassed and why
- How the data could still be accessed outside of VPD
- How to design VPD implementations
- How to protect VPD implementations

PeteFinnigan.com Limited
Oracle Security Expertise

Any Questions?

PeteFinnigan.com Limited
Oracle Security Expertise

Contact - Pete Finnigan

PeteFinnigan.com Limited
9 Beech Grove, Acomb
York, YO26 5LD

Phone: +44 (0) 1904 791188
Mobile: +44 (0) 7742 114223
Email: pete@petefinnigan.com