



**DOAG Konferenz
Mannheim 2006
Directors Cut ☺**

Rootkits in Oracle

Impressum

- ◆ **Oracle Schulung (SQL, DBA, PL/SQL, Security,...)**
- ◆ **Oracle Consulting & Support**
- ◆ **Oracle Entwicklung & Lizenzvertrieb**
- ◆ **Marco Patzwahl**
MuniQSoft GmbH
Grünwalder Weg 13 a
D-82008 Unterhaching b. München
- ◆ **Telefon: +49(0)89-67909040**
E-Mail: m.patzwahl@muniqsoft.de
Internet: www.muniqsoft.de



Definition

- ◆ **Rootkit-Programme manipulieren das Betriebssystem so, dass bestimmte Dateien, Registry-Einträge oder Prozesse für Systemprogramme wie Windows Explorer, Registrierungseditor oder Taskmanager unsichtbar werden.**
- ◆ **Damit ist es leicht, unbemerkt schädliche Codes/Programme auf dem PC einzuschleusen.**
- ◆ **Sony BMG machte sich diese Technik für seinen umstrittenen und mittlerweile zurückgezogenen XCP-Kopierschutz für CDs zunutze.**

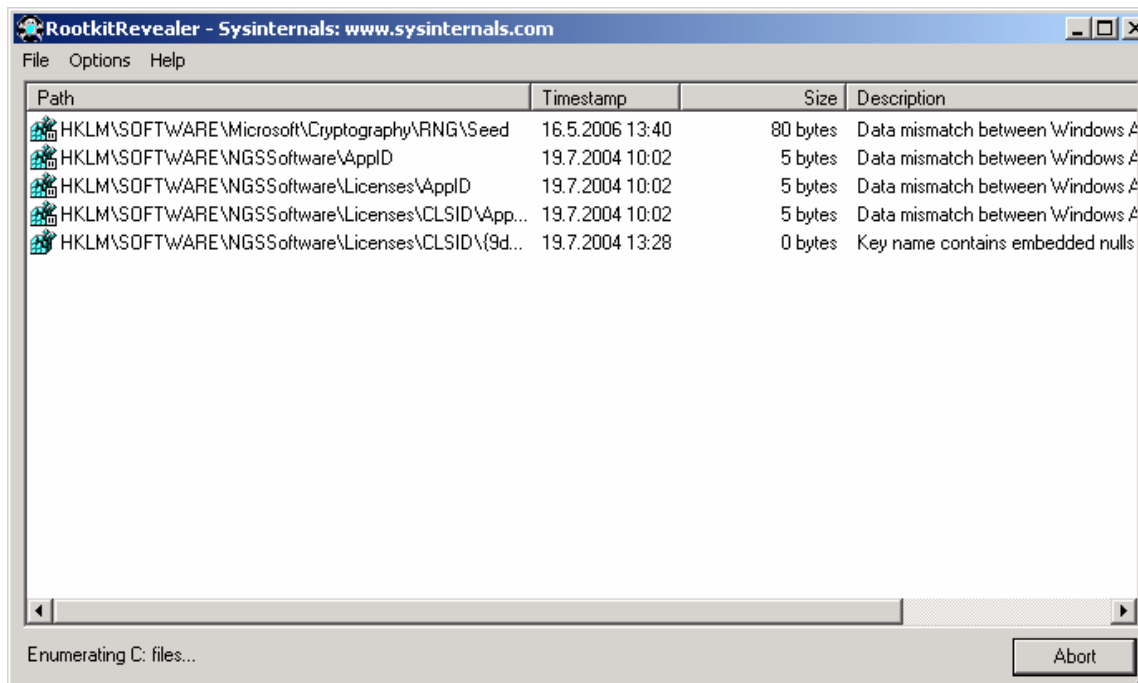
Rootkit unter Unix

- ◆ Die ersten Versionen modifizierten die Programme ps, ls und passwd um ihre Anwesenheit zu verbergen und ungehindert im System zu agieren.
- ◆ Damit konnten Tastatureingaben (Keylogger) mitgeschnitten oder Dateien unbemerkt ausgelesen oder modifiziert werden.

Rootkits unter Windows aufspüren

- ◆ Unter Windows kann von Sysinternals der Rootkitrevealer verwendet werden

▶ <http://www.sysinternals.com/Utilities/RootkitRevealer.html>



Typenunterscheidung

◆ Kernel Rootkits

- ▶ Teile des Betriebssystemcodes werden durch eigene Software ersetzt.
- ▶ Wird auch LKM (Loadable Kernel Modul genannt). Unter Windows kann man das z.B. durch .SYS Treiber realisieren.

◆ Userland Rootkits

- ▶ Wird häufig unter Windows eingesetzt, da es keinen Zugriff auf die Kernel-Ebene benötigt.
- ▶ Hier wird eine DLL benutzt, die z.B. über verschiedene Methoden (SetWindowsHookEx, ForceLibrary) Win-API Funktionen modifiziert und auf sich selbst umleitet.
- ▶ Damit können dann Ausgaben gefiltert werden.

Rootkit Beispiele

- ◆ **Hacker Defender (<http://hxdef.org/>) ist Open Source (HXDEF)**
- ◆ **Designerstudie Subvirt verfrachtet das Betriebssystem in eine virtuelle Maschine und ist dadurch nicht mehr auffindbar.**

Wer greift mich an?

◆ Extern:

- ▶ Script Kiddies
- ▶ Professionelle Hacker
- ▶ Konkurrenz?
- ▶ Terroristen

◆ Intern:

- ▶ Entlassene Mitarbeiter
- ▶ Verärgerte Mitarbeiter
- ▶ Mitarbeiter mit Liebeskummer ☹
- ▶ Bestochene Mitarbeiter (Konkurrenz, Geheimdienst, ...)

Was macht der Angreifer?

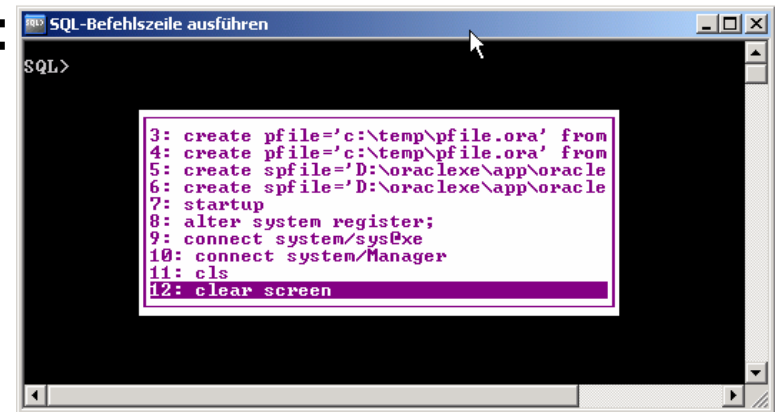
- ◆ **Daten auslesen (geheime, vertrauliche Unterlagen, Kreditkarten-Informationen, Gehälter, ...)**
- ◆ **Daten verändern (sehr schwer nachvollziehbar! Wie lange bewahren Sie Ihre Backups auf?)**
- ◆ **Daten löschen (evtl. incl. Backup), formatieren**
- ◆ **Von dort aus weitere Datenbanken/Rechner hacken**
- ◆ **Missbrauch als Datencontainer für illegale Software/Filme/Musik/Bilder!**
- ◆ **Verwendung des Rechners als Spam-Zombie, DoS Client**

Wie komme ich in die Oracle Datenbank?

- ◆ **Bekannte Benutzeraccounts/Passwörter versuchen**
- ◆ **Top 10**
 - ▶ **SYSTEM/MANAGER**
 - ▶ **SYS/CHANGE_ON_INSTALL AS SYSDBA**
 - ▶ **SYS/SYS AS SYSDBA**
 - ▶ **DBSNMP/DBSNMP**
 - ▶ **MDSYS/MDSYS**
 - ▶ **SCOTT/TIGER**
 - ▶ **OUTLN/OUTLN**
 - ▶ **RMAN/RMAN**
 - ▶ **HR/HR**
 - ▶ **CTXSYS/CTXSYS**

Wie komme ich in die Datenbank (f)

- ◆ **Keylogger auf den Client installieren. (Gibt es auch als Hardware zu kaufen)**
- ◆ **Passwörter in Skript-Dateien (Backupskripten, Monitoring-Skripten) suchen.**
- ◆ **Einschleusen von Code in Skript-Dateien (catalog.sql, catproc.sql, utlxplan.sql, utlrp.sql).**
- ◆ **Passwort in der History suchen:**
 - ▶ **Dos: <F7>**
 - ▶ **Unix: history**



The screenshot shows a window titled "SQL-Befehlszeile ausführen" with a black background and white text. The prompt "SQL>" is visible at the top left. A list of 12 commands is displayed in a white box with a purple border:

```
3: create pfile='c:\temp\pfile.ora' from
4: create pfile='c:\temp\pfile.ora' from
5: create spfile='D:\oracle\app\oracle
6: create spfile='D:\oracle\app\oracle
7: startup
8: alter system register;
9: connect system/sys@xe
10: connect system/Manager
11: cls
12: clear screen
```

Wie komme ich in die Datenbank (ff)

- ◆ **Brute Force Attack (Alle Passwort-Kombinationen versuchen)**
- ◆ **GLOGIN.SQL editieren:**
 - ▶ `GRANT DBA TO RSYS IDENTIFIED BY HACKER;`
- ◆ **Passwörter stehen bis Version 10.2 für Database Links im Klartext in Tabelle sys.link\$ Spalte password.**
- ◆ **Versenden von manipulierten SQL*Net Paketen (ohne Account) an die Datenbank mit eingeschleustem Code.**

Wie komme ich in die Datenbank (ff)

◆ Applikation besitzt DBA-Rechte ☹, deshalb

▶ SQL Injection bei dynamischem Cursor:

- `SELECT name FROM kunden where
/* kunden_id=1 */
kunden_id=1 UNION
SELECT username||'. '||password FROM
dba_users;`
- Danach mit Bruct-Force-Tool orabf das Passwort knacken.

Wie komme ich in die Datenbank (ff)

- ◆ Hacken des Listeners (z.B. Einschleusen von Codes, der einen Benutzer mit Administratoren-Rechten anlegt)
- ◆ Social Hacking
 - ▶ Passwörter erraten (Susi, Uschi, BMW, FC Bayern, ...)
 - ▶ Klebt ein Post-It auf dem Bildschirm mit dem Passwort, oder steht es unter der Tastatur?
 - ▶ USB-Sticks mit Trojaner in Firmennähe liegen lassen
- ◆ Phishing
 - ▶ Verschicken Sie Emails mit Text: Für Wartungsarbeiten auf den Database Ihr Password benötigt wird 😊

Wie komme ich in die Datenbank (ff)

- ◆ Wenn der Parameter `trace_level_client = 16` in `sqlnet.ora` Clientseitig gesetzt wurde, können SQL-Kommandos mitgelesen werden:
- ◆ ... npsend: 01 28 61 6C 74 65 72 20 |.(alter.)
- ◆ ... npsend: 75 73 65 72 73 20 73 79 |users.sy|
- ◆ ... npsend: 73 74 65 6D 20 69 64 65 |stem.ide|
- ◆ ... npsend: 6E 74 69 66 69 65 64 20 |ntified.|
- ◆ ... npsend: 62 79 20 6D 61 6E 61 67 |by.manag|
- ◆ ... npsend: 65 72 01 00 00 00 01 00 |er.....|

Ich bin drin ...

◆ Was nun ?

- ▶ Verwendung von diversen PL/SQL Packages, um Daten aus der Datenbank zu entwenden
- ▶ `utl_file` => schreibt auf lokale Platte
- ▶ `utl_tcp`, `utl_http` => verschickt per http/ftp/email
- ▶ `utl_smtp`, `utl_mail` => verschickt email
- ▶ `dbms_metadata`, ...). => zeigt Objektstrukturen an
- ▶ Freischalten von ftp oder http Ports mittels `dbms_xdb`

Namensauflösung bei Objekten

- ◆ **Wenn ein Objektname (Tabelle, View, Package, Procedure, Function) angegeben wird, geht Oracle wie folgt vor:**
 - ▶ **1. Gibt es im lokalem Schema ein Objekt mit diesen Namen (Achtung das lokale Schema lässt sich ändern mittels:**
 - `ALTER SESSION SET current_schema=<schema>;`
 - Wenn Nein,**
 - ▶ **2. Gibt es ein lokales Synonym mit diesem Namen**
Wenn Nein
 - ▶ **3. Gibt es ein Public Synonym mit diesem Namen**
Wenn Nein
 - ▶ **4. Fehlermeldung ausgeben (z.B. ORA-00942: Tabelle oder View nicht vorhanden)**

Beispieltricks

- ◆ **Legen Sie eine lokale View mit Namen all_tables für den Benutzer SCOTT an:**
 - ▶ `CREATE OR REPLACE VIEW scott.all_tables ...`
- ◆ **Legen Sie ein lokales Package utl_file für Scott an:**
 - ▶ `CREATE OR REPLACE package scott.utl_file`
- ◆ **Legen Sie ein lokales Synonym an:**
 - ▶ `CREATE SYNONYM utl_http ...`
- ◆ **Legen Sie ein globales Synonym an:**
 - ▶ `CREATE PUBLIC SYNONYM utl_tcp ...`

Manipulation der VerwaltungsvIEWS

- ◆ Ein Hacker muss nur in den üblicherweise verwendeten VerwaltungsvIEWS zusätzliche Filter einbauen, die die gewünschten Einträge verstecken.
- ◆ Da die meisten grafischen Tools nur die Inhalte von ALL_/DBA_ Objekten anzeigen, kann ein Hacker sich hier verstecken
- ◆ Wenn Sie herausfinden möchten auf welchen Tabellen eine VerwaltungsvIEW basiert, sehen Sie in dba_views oder noch besser in view\$ nach

Kurzübersicht Verwaltungstabellen/Views

DD Tabelle	DD View	Bemerkung
user\$	DBA_USER	Alle installierten Benutzer und Rollen
obj\$	DBA_OBJECTS	Alle installierten Objekte (Tabs/Procs/Trigger, ...)
tab\$	DBA_TABLES	Alle installierten normalen relat. Tabellen
col\$	DBA_TAB_COLUMNS	Alle Spalten der Tabellen
view\$	DBA_VIEWS	Alle installierten Views
trigger\$	DBA_TRIGGERS	Alle installierten Trigger
link\$	DBA_DB_LINKS	Alle Datenbank-Links
source\$	DBA_SOURCE	Alle Quellcodes (auch gewrappte)
syn\$	DBA_SYNONYMS	Alle Synonyme
ts\$	DBA_TABLESPACES	Alle Tablespaces

Manipulation der VerwaltungsvIEWS

◆ Beispiel: DBA_USERS

▶ Hinzufügen in der View:

▶ `CREATE OR REPLACE VIEW dba_users`

..

`AND username <> 'HACKER' ;`

▶ Blendet in der View nun 'Hacker' aus

◆ Alternative:

▶ Hacker arbeitet unter einem immer vorhandenen Account:

■ `SYS, SYSTEM, OUTLN, SYSMAN, XDB, MDSYS`

VerwaltungsvIEWS (V\$...)

- ◆ **GV\$ und V\$ Views sind Performance-Views, die nicht im üblichen Verwaltungsapparat (Tablespace SYSTEM) gespeichert werden, sondern z.B. im Controlfile.**
- ◆ **Sie basieren auch nicht auf einer View, sondern auf einer festen Struktur.**
- ◆ **Jedoch heißen die Originalobjekte V_\$... & GV_\$ und auf diesem liegt dann ein Public Synonym.**
- ◆ **Beispiel:**
 - ▶ **V_\$SESSION => Public Synonym mit Namen V\$SESSION**

VerwaltungsvIEWS (V\$...)

- ◆ Der Hackertrick besteht nun darin, eine lokale View mit Namen V\$... (z.B: V\$SESSION) anzulegen, in der ein gewünschter Filter gesetzt wird.
- ◆ Beispiel:
 - ▶ `CREATE OR REPLACE VIEW sys.V$$session
AS
SELECT * FROM sys.V_$session
WHERE username <> 'HACKER';`
 - ▶ `DROP PUBLIC SYNONYM v$session;`
 - ▶ `CREATE public synonym v$session
FOR sys.v$$session;`

VerwaltungsvIEWS (V\$...)

- ◆ **Alternativ wird einfach das Public Synonym auf ein anderes Objekt umgeleitet:**

- ▶

```
CREATE PUBLIC SYNONYM v$session
    FOR system.v$session_hack;
```

- ◆ **oder als lokales Synonym:**

- ▶

```
CREATE SYNONYM v$session
    FOR system.v$session_hack;
```

- ◆ **Hinweis:**

- ▶ **Zum Auslesen der View-Struktur verwendet man:**

- ▶

```
SELECT dbms_metadata.get_ddl
    ('VIEW', 'SYS', '<VIEW_NAME>')
FROM dual;
```

Abweichungen bei Benutzertabellen

- ◆ **Vergleichen Sie Original-Tabelle mit der DBA View:**
- ◆ **SELECT name FROM sys.user\$
minus
SELECT username FROM dba_users
minus
SELECT role FROM dba_roles;**
- ◆ **Ergebnismenge sollte nur die folgenden Benutzer ergeben:**
 - ▶ **PUBLIC**
 - ▶ **_NEXT_USER**

Abweichungen bei Triggern

- ◆

```
SELECT u.name,o.name
      FROM sys.trigger$ t, sys.user$ u, sys.obj$ o
      WHERE t.obj#=o.obj# and o.owner#=u.user#
      MINUS
      SELECT owner,trigger_name
      FROM dba_triggers;
```
- ◆ Die Liste sollte leer sein.

Abweichungen bei Jobs

```
◆ SELECT job, lowner, powner, what  
    FROM sys.job$  
    MINUS  
    SELECT job, log_user, priv_user, what  
    FROM dba_jobs;
```

◆ Die Liste sollte leer sein.

Abweichungen bei Objekten

- ◆

```
SELECT u.name,o.name
      FROM sys.obj$ o, sys.user$ u
      WHERE u.user#=o.owner#
            AND o.type# IN (7,8,9,11)
      MINUS
      SELECT owner,object_name
            FROM dba_objects
            WHERE object_type IN ('PROCEDURE',
                                   'PACKAGE', 'PACKAGE BODY', 'FUNCTION');
```
- ◆ Die Liste sollte leer sein.

Abweichung von Public Synonymen

```
◆ SELECT u.name "Syn.-Owner", o.name as "Syn.-
Name", s.owner, s.name as "ORIG.-NAME"
FROM sys.syn$ s, sys.obj$ o, sys.user$ u
WHERE o.owner#=u.user# AND s.obj#=o.obj#
AND o.name in
('V$SESSION', 'V$DATABASE', 'V$INSTANCE');
```

◆ Beispiel Ergebnismenge:

▶ PUBLIC	V\$INSTANCE	SYS	V_\$INSTANCE
▶ PUBLIC	V\$DATABASE	SYS	V_\$DATABASE
▶ PUBLIC	V\$SESSION	SYS	V\$\$SESSION
▶ SYSTEM	V\$DATABASE	SYSTEM	V\$LOGFILE

Eingeschmuggelte lokale Synonyme

- ◆

```
SELECT o.name, s.owner, s.name, s.node
FROM sys.syn$ s, sys.obj$ o
WHERE o.obj# = s.obj#
AND o.type# = 5
AND o.owner# = userenv('SCHEMAID')
AND (o.name LIKE 'V$%'
OR o.name LIKE 'USER_%'
OR o.name LIKE 'ALL_%'
OR o.name LIKE 'DBA_%'
OR o.name LIKE 'DBMS_%'
OR o.name LIKE 'UTL_%');
```
- ◆ **Als Benutzer SYSTEM (nicht SYS) oder beliebig anderer Benutzer, sollte diese Liste leer sein.**

Abweichung bei Privilegien

- ◆ **Leider ist eine Überwachung/Überprüfung hier sehr schwierig, denn System oder Objektrechte können entweder**
 - ▶ **Direkt**
 - ▶ **Über eine Rolle**
 - ▶ **Über eine Rolle die in einer Rolle liegt**
 - ▶ **Über eine Rolle die in einer Rolle die einer Rolle liegt**
 - ▶ **{Über ein Rolle}²⁵⁵**
- ◆ **zugewiesen werden**

SELECT führt DDL aus

- ◆ Sie waren bisher der Meinung, ein SELECT liest immer nur und verändert nichts??
- ◆

```
CREATE OR REPLACE FUNCTION ftest
RETURN NUMBER IS
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
EXECUTE IMMEDIATE 'create table t ( d date)';
RETURN 1;
END;
```
- ◆

```
CREATE OR REPLACE VIEW emp_view AS
SELECT * FROM scott.emp
WHERE 1=ftest;
```
- ◆

```
SELECT * FROM emp_view; /*Legt Tabelle t an*/
```

Objekte als Oracle eigene ausgeben

- ◆ **Welche der folgenden Objekte sind nicht von Oracle?**
 - ▶ **sys.dbms_backup_restore**
 - ▶ **sys.plitblm**
 - ▶ **sys.dbms_pickler**
 - ▶ **sys.dbms_util**
 - ▶ **sys.sys_stub_for_purity_analysis**
 - ▶ **sys.utl_idap**
 - ▶ **ctxsys.drue**
 - ▶ **ctxsys.drixml**
 - ▶ **sys.dbms_schema_name_export**

Lösung

- ◆ **Welche der folgenden Objekte sind nicht von Oracle?**
 - ▶ **sys.dbms_backup_restore (OK)**
 - ▶ **sys.plitblm (OK)**
 - ▶ **sys.dbms_pickler (OK)**
 - ▶ **sys.dbms_utl (Falsch)**
 - ▶ **sys.sys_stub_for_purity_analysis (OK)**
 - ▶ **sys.utl_idap (Falsch)**
 - ▶ **ctxsys.drue (OK)**
 - ▶ **ctxsys.drixml (Falsch)**
 - ▶ **sys.dbms_schema_name_export (Falsch)**

DDL Trigger

- ◆ Merken Sie sich die Liste der derzeit verwendeten DDL-Trigger.

```
◆ SELECT u.name, t.obj#, t.definition,  
        t.whenclause, t.action#  
        FROM sys.trigger$ t, sys.user$ u,  
        sys.obj$ o  
        WHERE t.obj#=o.obj#  
        AND o.owner#=u.user#  
        AND update$=0 AND insert$=0  
        AND delete$=0  
        ORDER BY name;
```

Wie kann man sich schützen?

◆ Gehen Sie bei Verdacht auf die Originaltabellen:

- ▶ `dba_users` → `sys.user$`
- ▶ `dba_tables` → `sys.tab$`
- ▶ `dba_objects` → `sys.obj$`
- ▶ `v$session` → `sys.v_$session`
- ▶ `v$process` → `sys.v_$process`

◆ Bei Packages, den Eigentümer voranstellen:

- ▶ `utl_file` → `sys.utl_file`
- ▶ `dbms_crypto` → `sys.dbms_crypto`

PL/SQL Codes entschlüsseln

- ◆ **Pete Finnigan hat einen Artikel zu diesem Thema verfasst ("How to unwrap PL/SQL")**
- ◆ **Damit können Root-Kits bzw. versteckte Funktionen in Standard-Packages eingeschmuggelt werden**
- ◆ **Sie können hier nur Prüfsummen oder die Größe der Packages vergleichen**

View Definitionen mit Hash-Wert speichern

- ◆ Speichern Sie für alle Views die Länge des View-Textes und einen Hash-Wert, drucken Sie die Liste aus und legen Sie diese in einen Tresor

- ◆ `SET SERVEROUTPUT ON SIZE UNLIMITED`

- ◆ `DECLARE /* als SYS starten */
v_text VARCHAR2(32000);
v_len NUMBER;`

`BEGIN`

`FOR c IN (SELECT owner,view_name,text FROM dba_views
WHERE owner='SYS' ORDER BY 2) LOOP`

`v_len:=length(c.text);`

`dbms_output.put_line(c.owner||'. '||rpad(c.view_name,30,'
')||' ('||lpad(v_len,4,' ')||') '||`

`dbms_crypto.Hash(utl_raw.cast_to_raw(c.text),
dbms_crypto.hash_md5));`

`END LOOP;`

`END;`

Beispielausgabe

◆	SYS.ALL_TAB_COLS	(6684)	69D3E30BA5A648E5374B9EB86DDEF30D
◆	SYS.ALL_TAB_COL_STATISTICS	(1985)	0B63DA6F08276D35366BEAEA7D6A7EC8
◆	SYS.ALL_TAB_COLUMNS	(525)	2D8F088AB0FDAE59486EF7A838616E
◆	SYS.ALL_TAB_COMMENTS	(1605)	7FDE414934207C71ECDD4416D9080D09
◆	SYS.ALL_TAB_HISTOGRAMS	(5491)	B0D4684558499CE808BCFFCDFE9CC607
◆	SYS.ALL_TABLES	(4691)	F43B61AB060F5E27F0EA9E2E8F1FA689
◆	SYS.ALL_TAB_MODIFICATIONS	(3137)	8CC0542B039FA72449ECF6C36807B71A
◆	SYS.ALL_TAB_PARTITIONS	(5804)	7EA4FAE6208376DE8E04B1199CD0F58F
◆	SYS.ALL_TAB_PRIVS	(559)	B94BB84E15C1E5829EC29856E73DA126

◆ ...

◆ **Bei Verdacht ein Angriffs, lassen Sie die Routine erneut laufen und vergleichen Sie die Ausgaben**

Rootkit Version 2

- ◆ **Vorgestellt von Alexander Kornbrust auf der Blackhat Konferenz 2006**
- ◆ **Er ersetzt in den Oracle Binaries alle Vorkommen der DD Tabelle user\$ durch einen anderen Namen z.B. usar\$**
- ◆ **Danach wird eine Kopie mit Namen user\$ erzeugt.**
- ◆ **Security-Scanner die jetzt dba_users mit user\$ vergleichen, finden nun keine Differenz, aber auch keinen Hacker☺**

Rootkit Version 2 aufspüren

◆ Wir suchen eine Tabelle, die die gleichen Spalten hat wie user\$

▶ Wir vergleichen hier nur die Spalte Default Tablespace (DATATS#), die kommt an 5ter Stelle nur in user\$ vor

▶ `SELECT owner#,name,ctime from obj$ WHERE obj# in (select c1.obj# from col$ c1, col$ c2 where (c1.name=c2.name and c1.col#=c2.col# and c1.name='DATATS#')and c1.obj#<>c2.obj#);`

▶

owner#	name	ctime
0	USER\$	29.06.05
0	USOR\$	18.11.06

Honeypot

- ◆ **Strategie um Hacker anzulocken.**
- ◆ **Es wird eine Datenbank oder ein Account absichtlich unzureichend geschützt um Hacker aufzuspüren.**
- ◆ **Beispiel:**
 - ▶ **SYSTEM/MANAGER Account wird die DBA Rolle entzogen**
 - ▶ **Audit für Create Session auf SYSTEM wird eingeschaltet**
 - ▶ **oder Logon Trigger verschickt eine Email an den DBA**

Oracle Würmer

- ◆ **Es existieren bereits Designer-Studien zu Oracle Würmern.**
- ◆ **Diese verschicken die gehashten Oracle-Passwörter an larry@oracle.com.**
- ◆ **Dieser Wurm ist noch relativ harmlos, aber was bringt die Zukunft?**

Security Tools / gute Webseiten

- ◆ **NGSSquirrel for Oracle ([http:// www.nextgenss.com](http://www.nextgenss.com))**
- ◆ **<http://www.petefinnigan.com/tools.htm> (kostenlos)**
- ◆ **AppDetective for Oracle (<http://www.appsecinc.com>).
Kosten: 900\$**
- ◆ **Projekt Lockdown**
 - ▶ **http://www.oracle.com/technology/pub/articles/project_lockdown/index.html**
- ◆ **<http://www.reddatabasesecurity.de/>**
- ◆ **<http://www.muniqsoft.de>**

Vorsichtsmaßnahmen

- ◆ **Alle Accounts prüfen und entweder gute/lange Passwörter definieren oder Account locken/Passwort zerstören.**
- ◆ **Listener mit Passwort schützen, Parameter `ADMIN_RESTRICTIONS_{listener_name}=ON` in listener.ora setzen**
- ◆ **CREATE DATABASE LINK Recht aus der Connect Rolle entfernen (ab 10.2 bereits durch Oracle erledigt).**
- ◆ **EXECUTE Recht auf `utl_tcp`, `utl_file` und `utl_inaddr` von Public entfernen.**
- ◆ **GLOGIN.SQL Datei schützen, denn diese wird bei jedem Start von SQL*Plus ausgeführt (ab 10g bei jedem Neuconnect).**

Zusammenfassung

- ◆ **Schützen Sie sich vor Angriffen aus dem Internet oder Intranet.**
- ◆ **Verbessern Sie die Sicherheit Ihres Betriebssystems und der Hardware (USB-Ports, Floppy-LW, ...).**
- ◆ **Optimieren Sie die Sicherheit für die Oracle Datenbank (Accounts, Listener, Hash-Werte für Views, Logik-Prüfungen, ...)**
- ◆ **Gehen Sie in eine Security-Schulung (z.B. zur Firma MuniQSoft)**
- ◆ **und ...**
 - ▶ **alles wird gut ☺**

Impressum

- ◆ **Oracle Schulung (SQL, DBA, PL/SQL, Security,...)**
- ◆ **Oracle Consulting & Support**
- ◆ **Oracle Entwicklung & Lizenzvertrieb**
- ◆ **Marco Patzwahl**
MuniQSoft GmbH
Grünwalder Weg 13 a
D-82008 Unterhaching b. München
- ◆ **Telefon: +49(0)89-67909040**
Fax: +49(0)89-67909050
E-Mail m.patzwahl@muniqsoft.de
Internet: www.muniqsoft.de

