Summary:

Self developed Oracle Reports are vulnerable against SQL Injection if these reports are using “lexical references” without input validation. Most Oracle reports developers are not aware of this problem and are not validating the input (e.g. from parameters) in Oracle Reports. As every input validation bug it is not a problem of the development tool itself (in this case Oracle Reports Developer) it is a problem of the developers using Oracle Reports developer. The Oracle documentation holds back information about this potential problem. Large quantities of Oracle Reports are vulnerable against SQL Injection.

About Oracle Reports:

Oracle Reports is Oracle's award-winning, high-fidelity enterprise reporting tool. It enables businesses to give immediate access to information to all levels within and outside of the organization in an unrivaled scalable and secure environment. Oracle Reports consists of Oracle Reports Developer (a component of the Oracle Developer Suite) and Oracle Application Server Reports Services (a component of the Oracle Application Server). The Oracle E-Business Suite is also using Oracle Reports.

Affected products:

All generated Oracle reports using lexical reference since Oracle Reports 2.0. These reports could be self developed or be part of an Oracle application (e.g. E-Business Suite).

Fix:

It is not possible to disable the “lexical references” functionality by setting a special environment setting. It is necessary to fix this problem in every report by validating every parameter in an After-Parameter-Form-Trigger.

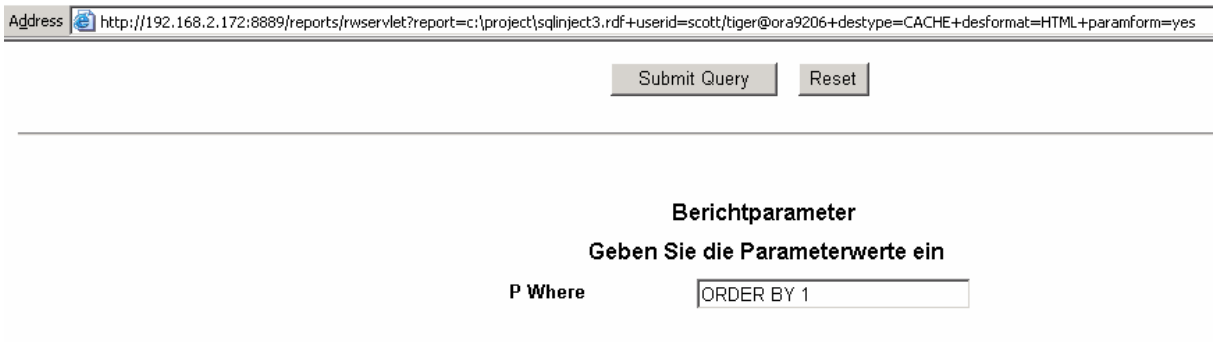
Background:
Oracle Reports are created with the Oracle Reports developer and are quite common in the enterprise environment. Oracle itself is using Oracle Reports e.g. in their E-Business-Suite.

Oracle Reports provides a feature called *lexical references*. A lexical reference is a placeholder for text that you embed in a SELECT statement. It is possible to replace the clauses appearing after SELECT, FROM, WHERE, GROUP BY, ORDER BY, HAVING, CONNECT BY and START WITH.

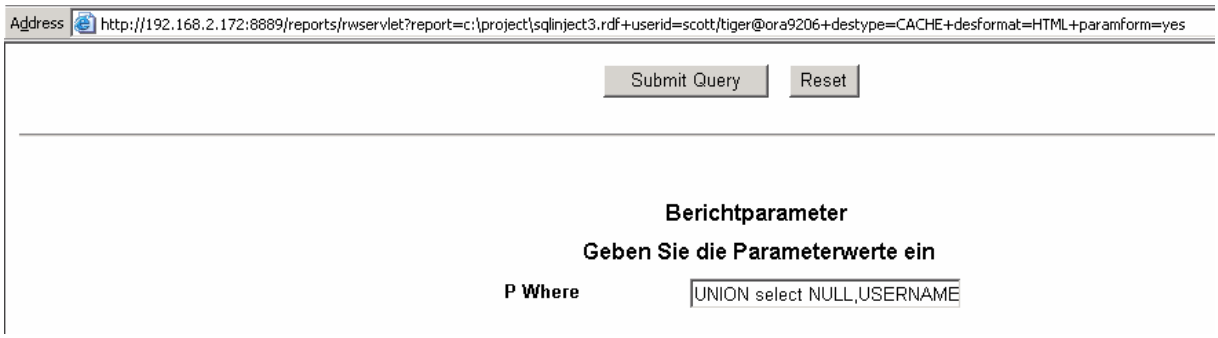
Short demonstration of SQL Injection in Oracle Reports
The following vulnerable sample report for the demo user scott/tiger can be downloaded from http://www.red-database-security.com/wp/demo_sql_injection_reports.zip . To run this report an Oracle Reportsserver is required.

1. Run an Oracle Reports via a web browser
(e.g.
http://myserver:8889/reports/rwservlet?report=sqlinject3.rdf+userid=scott/tiger@ora9206+destype=CACHE+desformat=HTML)

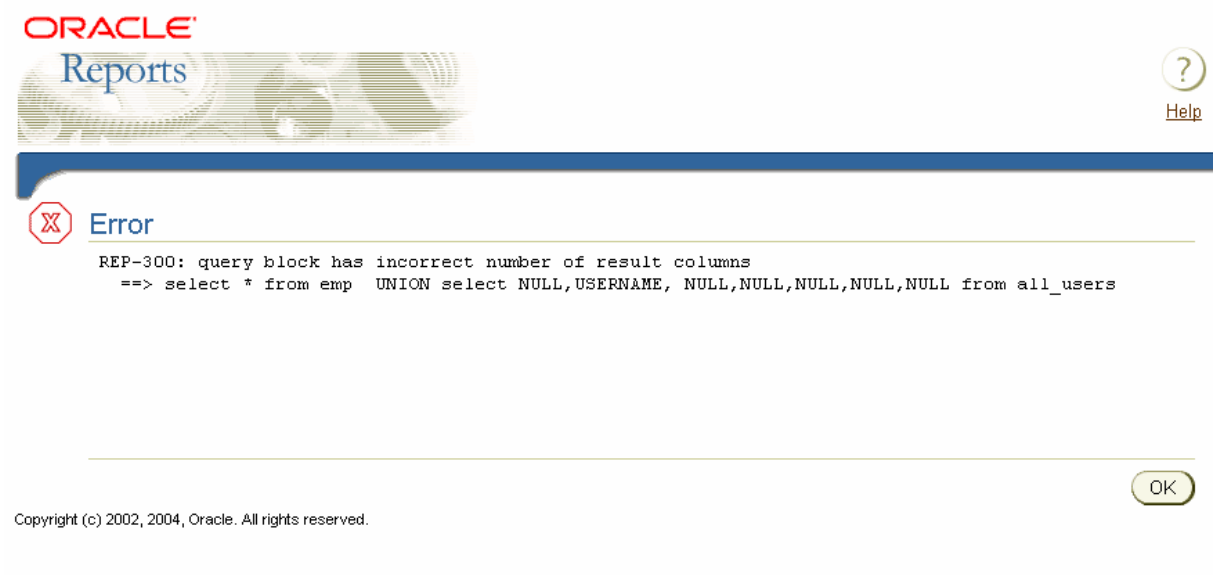
2. Add the parameter ***paramform=yes*** to the URL and resubmit the URL again. A HTML window appears which allows a user to modify parameter values from a web page, e.g. change the sort sequence (e.g. ORDER BY ENAME)



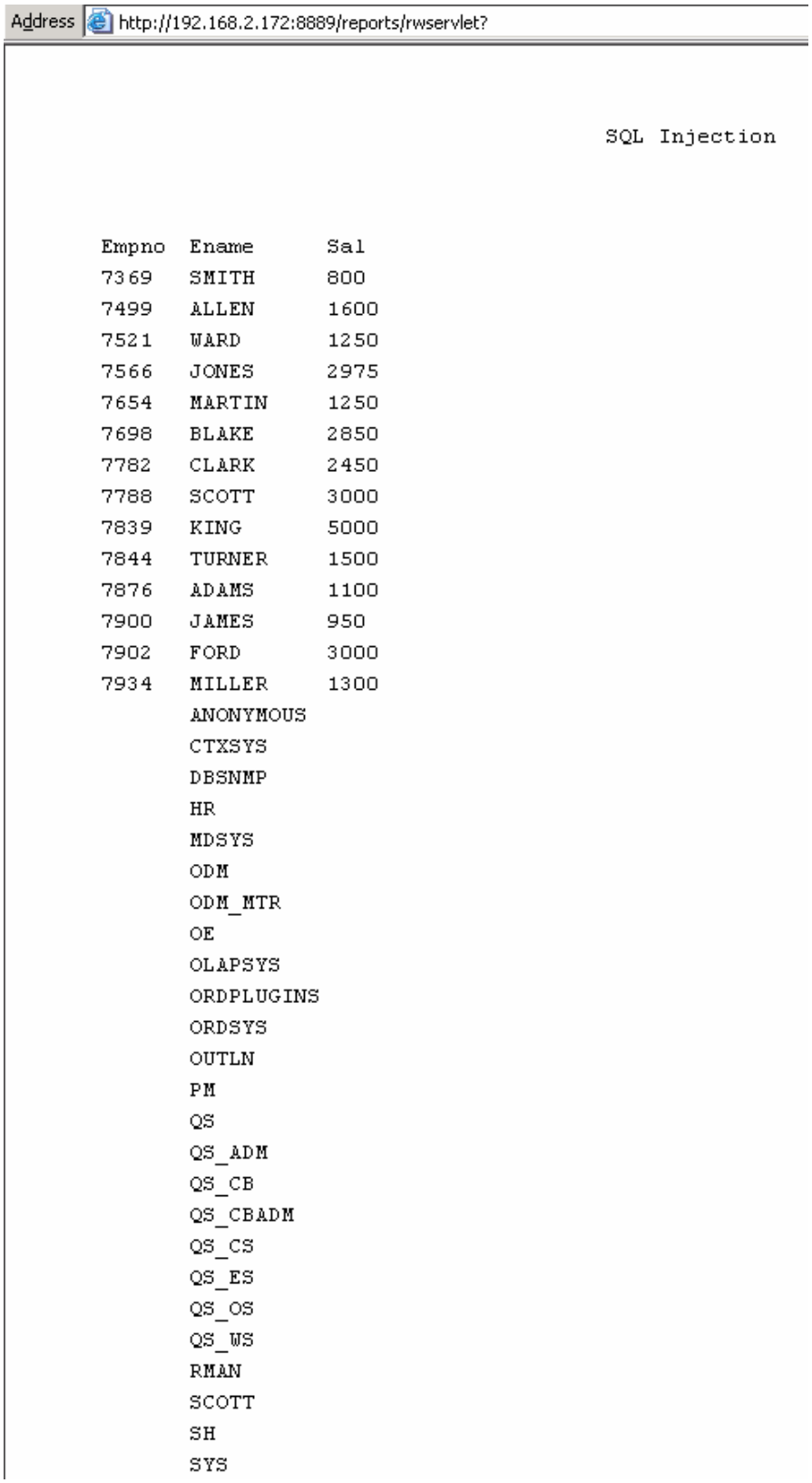
3. Replace the default value "***ORDER BY 1***" of the parameter P_WHERE with the string "***UNION select NULL,USERNAME, NULL,NULL,NULL,NULL,NULL,NULL from all_users***"



If the resulting SQL statement is not correct Oracle reports returns the appropriate error message (e.g. REP-300)



4. Submit the modified query
Oracle Reports server replaces the parameter P_WHERE with the value submitted by the URL and executes the statement.



Impact:

Lexical references are a powerful and quite common feature in Oracle Reports that's why a good and flexible way to parameterize Oracle reports. Large enterprise customers have sometimes hundreds different reports. Every report should be checked for this vulnerability.

The impact of this generic SQL injection depends on the permission of your Oracle Reports user. Too many privileges (e.g. DBA privilege) could expose the hashkey of Oracle database users. If the Oracle password is too short or weak it is possible to get the plaintext password with special tools (e.g. 150.000 pw/sec with Checkpwd).

I haven't tested the E-Business-Suite for this SQL Injection vulnerability.

Fix:
It is not possible to disable the "lexical references" functionality by setting a special environment variable. It is necessary to fix this problem in every report by validating every parameter in an After-Parameter-Form-Trigger. This is can be time consuming task if you check several hundreds of reports

References:

- Metalink Document 115072.1: Complete Resource Reference for using Lexical Parameters in Oracle Reports
- Oracle Password Checker: Checkpwd 1.1

History:

- 13-may-2004 Oracle secalert was informed to give Oracle time to fix possible issues in their own reports (e.g. in the E-Business-Suite)

Other Oracle security related documents:

Hardening Oracle Application Server 9i Rel.1, 9i Rel.2 and 10g:
http://www.red-database-security.com/wp/DOAG_2004_us.pdf

SQL Injection in Oracle Forms
http://www.red-database-security.com/wp/sql_injection_forms_us.pdf

Oracle security training:

http://www.red-database-security.com/security_training/oracle_anti_hacker_training.html

Customized inhouse-security trainings available.

About Red-Database Security GmbH:
Red-Database-Security GmbH is a specialist in Oracle Security. We are offerings Oracle security trainings, database and application server audits, penetration tests, oracle (security) architecture reviews and software security solutions against Oracle rootkits.

Contact:
If you have questions or comments you could contact us via

info at red-database-security.com