

WaveFront Consulting Group
BLIND SQL INJECTION
 (UBC)

Rui Pereira, B.Sc. (Hons), CISSP, CIPS ISP, CISA, CWNA, CPTS/CPTE
 WaveFront Consulting Group Ltd
ruiper@wavefrontcg.com
www.wavefrontcg.com

© WaveFront Consulting Group

This material is Copyrighted © by the WaveFront Consulting Group Ltd, and must not be used, copied or otherwise distributed, in whole or in part, without the express written permission of the author(s).

Please contact ruiper@wavefrontcg.com if you have any doubts about the copyright restrictions applicable to this publication.

© WaveFront Consulting Group

WHO AM I

- Rui (Roy) Pereira
- Over 25 years in Information Technology industry
- Last 13 years specializing in Information Security and Audit
- CISSP and CISA certified, CIPS ISP, CWNA, CPTS/CPTE
- Independent consultant
- Specializing in...
 - Threat-Risk Assessments, Investigations, DRP/BCP
 - Penetration Testing and Vulnerability Assessments
 - Policy Development, Security Awareness and Education
 - Applications Security, PCI/SOX Compliance
 - Security Architecture Design and Implementation
 - Wireless and Telecommunications Security
- Customers include BC and Alberta Governments, E-Comm, ICBC, LDB, Citadel Commerce, Top Producer, etc.
- Teach various courses at BCIT and UBC, including Intro to Computer Crime and Wireless Network Security

© WaveFront Consulting Group

AGENDA

- Database Attacks
- Verbose SQL Injection
- Blind SQL Injection
- An Example
- Finding Blind SQL Injection
 - W3AF, Samurai, SQLMap
- Fixing SQL Injection

© WaveFront Consulting Group

DATABASE ATTACKS

- Listed as one of SANS Top 20 Security Risks for 2007
 - <http://www.sans.org/top20/#s7>
- Indirect Attacks
 - Attack the application which uses the database
 - SQL Injection (Verbose and Blind)
- Direct Attacks
 - Exploit vulnerabilities in database system and underlying OS
 - Buffer overflows, weak passwords or access control, lack of auditing, vulnerable services, etc.
 - Includes password guessing, cracking, stealing
 - Includes bypassing database and accessing underlying disk file/storage structures

© WaveFront Consulting Group

DIRECT DATABASE ATTACKS

- SQL Slammer/Saphire Worm hit in January 2003
 - Exploited a buffer overflow vulnerability in Microsoft's SQL Server and MSDE 2000
 - Infected more than 90% of vulnerable hosts in 10 mins
 - www.caida.org/publications/papers/2003/sapphire/sapphire.html
 - www.microsoft.com/technet/security/bulletin/MS02-039.msp
 - Exploit in Metasploit Framework 3 exploits/windows/mssql/ms02_039_slammer.rb (also is MSF2)
- Other exploits exist in MSF2&3, and elsewhere
 - SQL 2K HELO Overflow, www.securityfocus.com/bid/5411
 - SQL 2K5 sqldmo.dll ActiveX Buffer Overflow, <http://www.securityfocus.com/bid/25594>

© WaveFront Consulting Group

SQL SERVER

- If trusted connections are used, then OS access may be sufficient for database access
 - Trusted connections use Windows accounts to access the database
- Blank sa admin accounts are a problem
 - With Oracle problem is huge number of seeded accounts
- If xp_cmdshell and similar procedures are available, can access OS functionality
 - Even if xp_cmdshell is not available, can be re-enabled with sp_configure
- Can enter multiple SQL commands together on same line

© WaveFront Consulting Group

DATABASE COUNTERMEASURES

- Enforce strong password policies
 - Change default account passwords
 - Oracle has over 600 of these, SQL Server has sa
- Patch database and underlying OS
 - Setup ongoing patch management process
- Harden (secure) database and OS installs
 - Remove dangerous xp_ and sp_ procedures, or restrict access to them (SQL Server)
- Code applications securely to reduce risk of SQL Injection
 - Architect web applications to reduce risk of compromise, impact should compromise occur
- Restrict access to database ports via internal and external network and host-based firewalls

© WaveFront Consulting Group

SQL INJECTION

- Attacking the application which uses the database
 - Attacker includes SQL statements or fragments as input to the application
 - Application accepts this input and passes it, as is, to the backend database for processing
 - Application does not validate client input
 - Allows authentication bypass, unauthorized access to data (crud), access to OS system commands
 - Example: <http://www.testfire.net/bank/login.aspx>
 - Finding potentially vulnerable sites using Google
 - [inurl:"id=10"](#) (Michael Sutton's Blog)

© WaveFront Consulting Group

SQL INJECTION

Normal Query

```
SELECT COUNT (*) FROM User.tbl
WHERE UserName='Rui'
AND Password='IamaWalrus'
```

Malicious Query

```
SELECT COUNT (*) FROM User.tbl
WHERE UserName='' OR 1=1--
AND Password=''
```

"OR 1=1" is always true, matches every record in table

"--" comments out rest of query

© WaveFront Consulting Group 10

WHAT CAN AN ATTACKER DO?

- Bypass authentication and access controls
- Determine structure of backend database
 - Database enumeration
- Unauthorized access to data
 - Including passwords and credit card numbers if stored in plaintext
- Unauthorized changes to data
 - Including deletion of records and tables, insertion of records
- Able to run Operating System commands

© WaveFront Consulting Group 11

SQL INJECTION

- Verbose (Normal) SQL Injection
 - Attacker relies on SQL error messages to piece together the SQL code being executed, enumerate backend database tables, columns, etc.
- Blind SQL Injection
 - Application does not return SQL error messages
 - Its behavior when subjected to erroneous input is used to detect SQL Injection possibilities
 - We ask the server a series of true/false questions and build up our results from the answers

© WaveFront Consulting Group 12

VERBOSE SQL INJECTION

- Error message
 - The close brackets “)” were provided on input

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
([Microsoft][ODBC Microsoft Access Driver]

Extra ) In query expression 'Username='' AND Password ='
/_employees/login3.asp, line 49
```

© WaveFront Consulting Group



BLIND SQL INJECTION

- These statements don't generate errors, give same result:

```
http://www.acme.com/pressRels.jsp?pressRelID=5 AND 1=1
http://www.acme.com/pressRels.jsp?pressRelID=4+1
```

- These don't generate errors (or return nothing):

```
http://www.acme.com/pressRels.jsp?pressRelID=5 AND 1=2
http://www.acme.com/pressRels.jsp?pressRelID=5 AND
  USER_NAME() = 'dbo'
```

- But this statement returns press release #5:

```
http://www.acme.com/pressRels.jsp?pressRelID=5 AND
  USER_NAME() = 'sa'
```

- So we know the current user name is sa!

© WaveFront Consulting Group



BLIND SQL INJECTION

- In practice tests are done character by character
 - A binary search is used to speed things up
- `http://.../...jsp?pressRelID=5 AND ascii(lower(substring((SELECT TOP 1 name FROM sysobjects WHERE xtype='U'), 1, 1))) > 109`
 - If the fifth press release is returned, then first letter comes after 'm'
- `http://.../...jsp?pressRelID=5 AND ascii(lower(substring((SELECT TOP 1 name FROM sysobjects WHERE xtype='U'), 1, 1))) > 116`
 - If the fifth press release is not returned, then letter is greater than ASCII 109 ('n') and less than 116 ('t')
 - And so on...

© WaveFront Consulting Group



BLIND SQL INJECTION

- A secure application would reject such requests because it treats the user's input as a value
 - The value "5 AND 1=2" would cause a type mismatch error (if passed to a stored procedure)
 - The server would not display a press release (may display generic or custom error message)
 - Simply disabling the display of database error messages does not offer sufficient protection against SQL injection attacks

TYPES OF BLIND SQL INJECTION

- Conditional Response
 - SELECT field FROM table WHERE otherfield='value' AND 1=1 produces normal page
 - SELECT field FROM table WHERE otherfield='value' AND 1=2 produces different page
- Conditional Error
 - SELECT 1/0 FROM table WHERE field='value' will generate error if field='value'
- Time Delays
 - SQL engine executes a long running query or a time delay statement depending on the logic injected
 - Attacker can measure how long page takes to load to see if injected statement is true

BLIND SQL EXAMPLE

- WordPress is Open Source Blogging software
 - Blind SQL Injection in WordPress 2.1.3
 - Advisory at <http://www.waraxe.us/advisory-50.html>
 - Exploit at <http://www.waraxe.us/ftopic-1776.html>
 - Uses time delays to get hash of WordPress password one character at a time
 - Can then brute force hash (no salt), or create valid session cookie from hash

FINDING SQL INJECTION

- Manual Tests
- Automated Scanners
 - HP WebInspect
 - IBM Rational Appscan
 - Cenzi Hailstorm
 - Syhunt Sandcat
 - Paros Proxy
 - W3AF
 - etc...
- SQL Injection Tools
 - SQLMap
 - SQL Ninja (SQL Server)
 - SQLix
 - etc...

© WaveFront Consulting Group



FIXING SQL INJECTION

- Validate all user input to ensure it is appropriate for the particular information the field represents
 - User names do not usually contain semicolons (;), tildes (~) or ampersands (&)
 - May contain single quotes though (O'Brien)
 - In general, limit all client-supplied data to alpha-numeric characters whenever possible
- Filter out potentially dangerous characters
 - Major culprits are the single quote ('), double dash (--), and semicolon (;)
 - Remove these entirely
 - Replace them with 'safe' equivalents
 - Use HTML encoding like &code; or &#number;
 - Other characters that can be used for SQL Injection include " , \ \ * & () \$ % ^ @ ~ ?

© WaveFront Consulting Group



FIXING SQL INJECTION

- Filtering and validation should be part of a global facility called when any page is loaded / submitted
 - It should be done on all pages and fields
 - But see subsequent section on data validation and on black vs. white listing
- Use language platform tools to handle potentially dangerous characters
 - Such as cfqueryparam in ColdFusion
- Do not create dynamic SQL queries using simple string concatenation
- Replace individual single quotes in string data with two single quotes
- Ensure numeric variables contain numeric data

© WaveFront Consulting Group



FIXING SQL INJECTION

- Run the database and web server under low privilege accounts
 - Do not access database from Web server using dba or database owner accounts
 - Do not run database under privileged system account
 - Do not run Web Server as root or administrator
 - Do not allow database server to initiate outbound access (use internal firewalls)
- This does not prevent SQL Injection, but reduces the impact of such attacks

© WaveFront Consulting Group



22

FIXING SQL INJECTION

- Use parameterized queries (.NET), prepared statements (Java)
 - Use strongly typed parameters
- Use stored procedures to abstract data access, so users do not directly access tables and views
 - Access control can then be applied to stored procedures instead of the underlying tables and views
 - NEVER take input from client and add it verbatim to SQL statements
 - Do not pass SQL statements, or fragments of SQL statements, as parameters to stored procedures
 - Stored procedures (should) prevent attacker from providing SQL code to be executed
 - Parameters are treated as literal values, not as executable code

© WaveFront Consulting Group



23

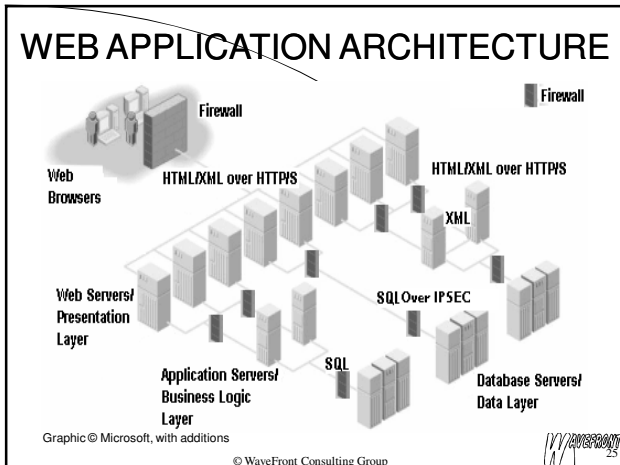
FIXING SQL INJECTION

- Turn off verbose SQL error messages
 - Can still use Blind SQL Injection
- Improve application level filters to better detect these kinds of attacks
 - mod_security on Linux Apache Web Servers
 - Can be deployed as a separate server for Windows IIS shops
- Use web application firewalls and Intrusion Detection / Prevention Systems (IDS/IPS)

© WaveFront Consulting Group



24



- ### DATABASE SCANNERS
- Work with various database systems
 - Oracle, SQL Server, MySQL, DB2, Sybase
 - Look for configuration issues, as well as vulnerabilities
 - AppDetectivePro
 - www.appsecinc.com/products/appdetective
 - Audit and pen-test components
 - Integrity AppSentry
 - www.integrigy.com/products/appentry
 - Tests Oracle e-Business Suite, Web Application Server
 - SCUBA (free)
 - www.imperva.com/products/scuba.html
 - More of an audit tool
 - Nessus has several Oracle and SQL Server checks
- © WaveFront Consulting Group

- ### SQL INJECTION REFERENCES
- Advanced SQL Injection in SQL Server Applications, http://www.nextgens.com/papers/advanced_sql_injection.pdf
 - More Advanced SQL Injection, http://www.nextgens.com/papers/more_advanced_sql_injection.pdf
 - SQL Injection Walkthrough, <http://www.securiteam.com/securityreviews/5DP0N1R76E.html>
 - SQL Injection Attacks by Example, <http://www.unixwiz.net/techtips/sql-injection.html>
 - SQL Injection Whitepaper, <http://www.spidynamics.com/papers/SQLInjectionWhitePaper.pdf>
 - SQL Injection: Finding and Fixing It, <http://www.acunetix.com/websec/sql-injection.htm>
 - Blind SQL Injection, http://www.spidynamics.com/whitepapers/Blind_SQLInjection.pdf and www.imperva.com/resources/adc/blind_sql_server_injection.html
 - SQL Security Website, <http://www.sqlsecurity.com/>
- © WaveFront Consulting Group

SQL INJECTION REFERENCES

- The Database Hacker's Handbook: Defending Database Servers, David Litchfield / Chris Anley / John Heasman / Bill Grindlay, Wiley, 2005
- How Prevalent Are SQL Injection Vulnerabilities?, <http://portal.spidynamics.com/blogs/msutton/archive/2006/09/26/How-Prevalent-Are-SQL-Injection-Vulnerabilities-3F00.aspx>
- SQL Injection Cheat Sheet, <http://ferruh.mavituna.com/makale/sql-injection-cheatsheet/>
- MS-SQL Injection Cheat Sheet (and others), <http://pentestmonkey.net/blog/mssql-sql-injection-cheat-sheet/>

© WaveFront Consulting Group



Thank You

Rui Pereira, B.Sc.(Hons), CISSP, CIPS, ISP, CISA, CWNA, CPTS/CPTE
WaveFront Consulting Group Ltd
604 961 0701
ruiper@wavefrontcg.com
www.wavefrontcg.com

© WaveFront Consulting Group