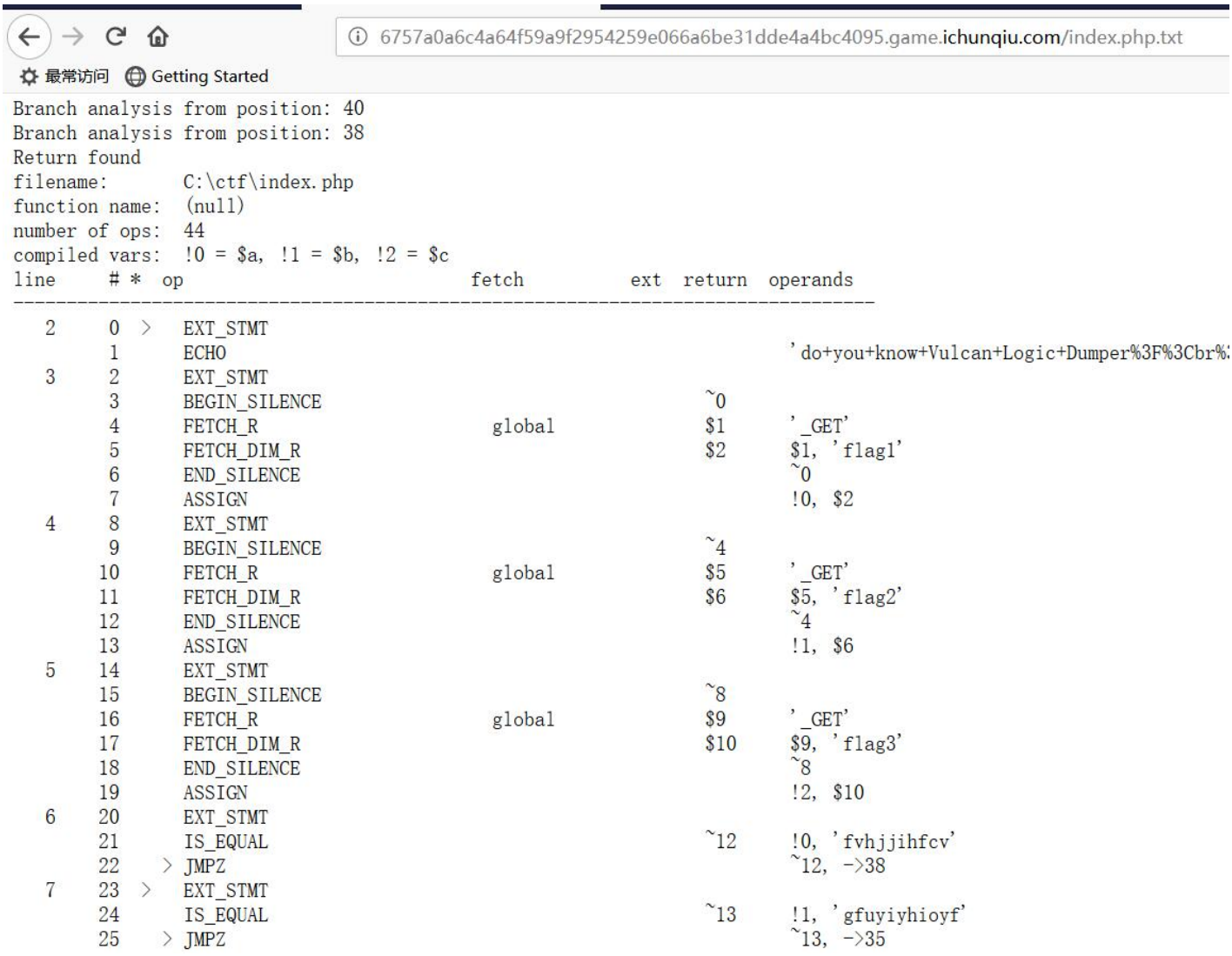
i春秋——“百度杯”CTF比赛 十月场——Vld（Vulcan Logic Dumper 、php opcode、sql 报错注入）...

转载

weixin_30553837 于 2018-10-14 00:13:00 发布 359 收藏
原文链接：http://www.cnblogs.com/leixiao-/p/9784904.html

打开题目看到提示 "do you know Vulcan Logic Dumper?" ，再查看源码看到"<!-- index.php.txt ?>"，访问后发现一堆看不懂的东西




```
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+Dumper%3F%3Cbr%
   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
```

这肯定就是所谓的Vulcan Logic Dumper了，先了解下相关概念

PHP内核-Zend引擎：http://www.php.cn/php-weizijiaocheng-355597.html

PHP中的opcode：https://blog.csdn.net/weiyuanke/article/details/76921476

Vulcan Logic Dumper：http://www.phppan.com/2011/05/vld-extension/

也就是说我们刚才看到的一堆代码其实就是借助vld得到的，php语言中供zend引擎执行的中间代码opcode。有了opcode便可以将其翻译成php代码。 网上也没找到翻译opcode的工具，只好借着对照表自己人工翻译了...

(opcode对照表：http://www.php.net/manual/en/internals2.opcodes.list.php)

这段代码比较简单，其实掌握下规律还是挺好分析的，这是我初步分析的结果

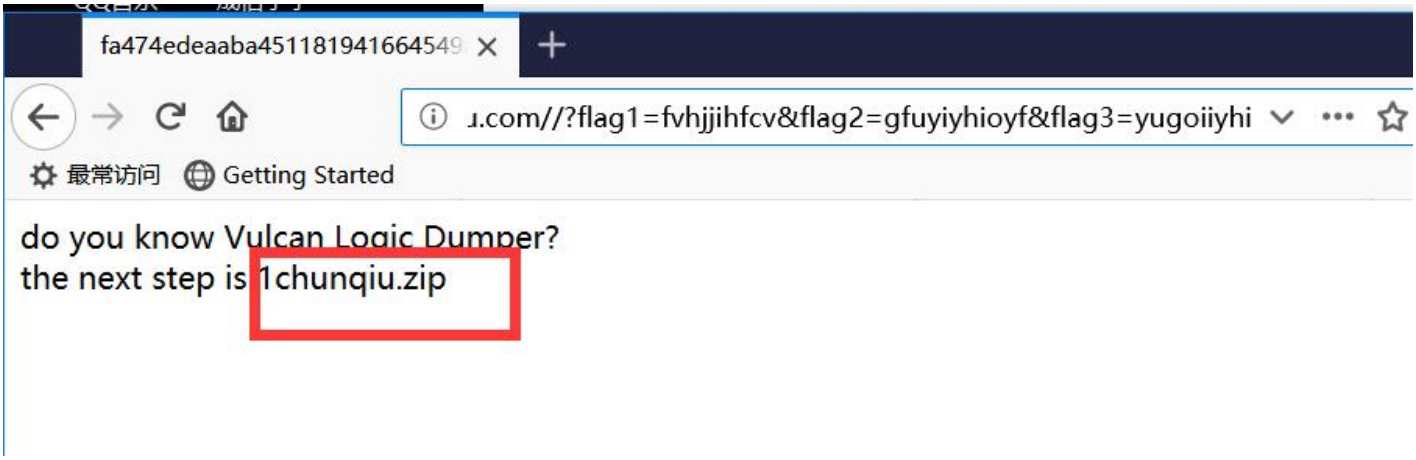
```
<?php
    echo'do+you+know+Vulcan+Logic+Dumper%3F%3Cbr%3E';


    $0=$_GET['flag1'];


    $1=$_GET['flag2']


    $2=$_GET['flag3'];



        21      如果$0不等于'fvhjjihfcv'
        22      跳转到38行


        24      如果$1不等于'gfuyiyhioyf'
        25      跳转到35行


        27      如果$2不等于'yugoiiyhi'
        28      跳转到32行


        30      echo'the+next+step+is+xxx.zip';

        31      跳转到34行

        32      EXT_STMT

        33      echo'false%3Cbr%3E';

        34      跳转到37行

        35      EXT_STMT

        36      echo'false%3Cbr%3E';

        37      跳转到40行

        38      EXT_STMT

        39      echo'false%3Cbr%3E';

        40      NOP
        41      EXT_STMT

        42      echo%3C%21--+index.php.txt+%3F%3E%0D%0A%0D%0A';
?>
```

进一步转换为php代码则为

```php
<?php

    echo 'do you know Vulcan Logic Dumper?<br>';
    $a=$_GET['flag1'];
    $b=$_GET['flag2'];
    $c=$_GET['flag3'];

    if($a!='fvhjjihfcv')
    {
        echo 'false<br>';
    }
    elseif($b!='gfuyiyhioyf')
    {
        echo 'false<br>';
    }
    elseif($c!='yugoiiyhi')
    {
        echo 'false<br>';
    }
    else
    {
        echo 'the next step is xxx.zip';
    }


    echo '<!-- index.php.txt ?>';
?>
```

代码很简单，不用多解释

构造/?flag1=fvhjjihfcv&flag2=gfuyiyhioyf&flag3=yugoiiyhi



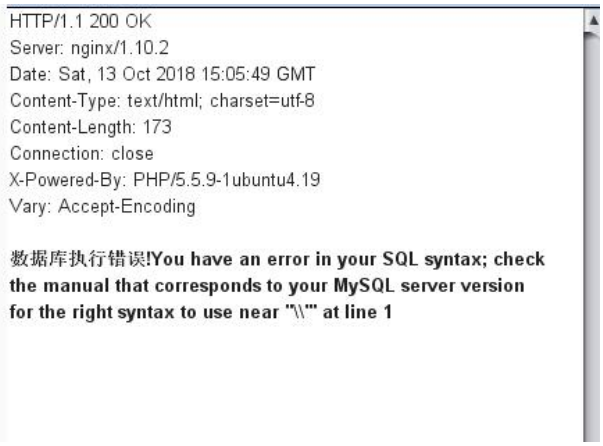
接着访问/1chunqiu.zip，下载完文件后解压，开始代码审计。

password经过md5加密，number只能是纯数字，所以都不存在注入点。但是username虽然经过addslashes()处理（单引号，反斜杠等前面都会被加上反斜杠而转义，防御sql注入），但是又再次被这句代码处理 " $username = trim(str_replace($number, '', $username)); "，所以我们可以利用这里让单引号逃逸出来，这句代码来的很突兀，而且没什么意义，很明显故意的漏洞。

```php
$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));
```

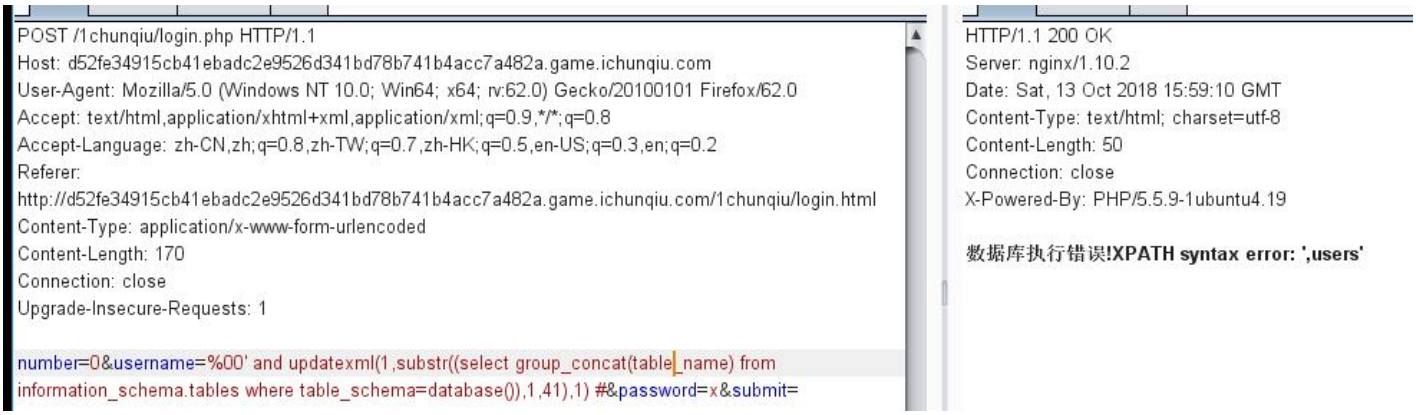
```php
public function safe_data($value){
    if( MAGIC_QUOTES_GPC ){
        stripcslashes($value);
    }
    return addslashes($value);
}
```

构造number=0&username=%00' &password=3

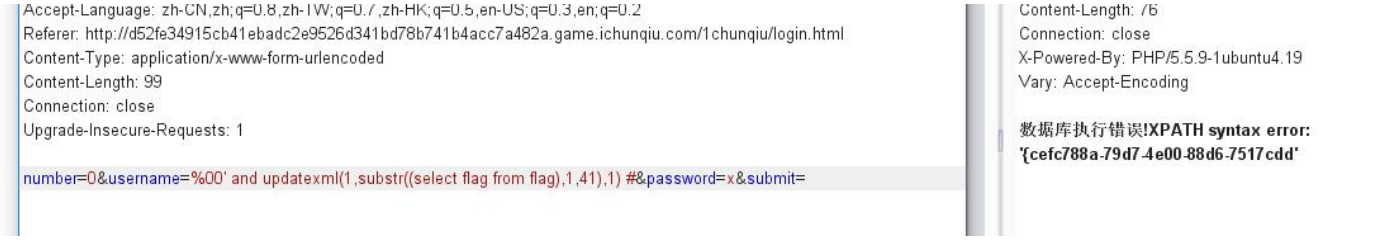


看到数据库报错，说明单引号逃逸成功，当username提交 %00' ，经过addslashes()处理后（addslashes()会在NULL前加 \ ,0等于NULL）是 \0\'。而number也是0,所以将从username中去掉0，username则变成 \\' ，单引号前的\被\转义，所以单引号逃逸成功，后台sql语句为 select * from`users`where username=' \\ ' ' ，可见多出一个单引号，当然报错。

由于没有数据回显点，所以考虑进行报错注入,number=0&username=%00' and updatexml(1,substr((select group_concat(table_name) from information_schema.tables where table_schema=database()),1,41),1) #&password=x&submit=



测试多次，无论表名列名都有4个字符不显示，本来以为服务器会过滤掉这段4个字符的字符串，但是尝试部分截取也还是不显示。。。 猜到是flag，所以就直接查询了



注意语句中不要再出现0了，，被坑了好久，最后才发现。。

转载于:https://www.cnblogs.com/leixiao-/p/9784904.html