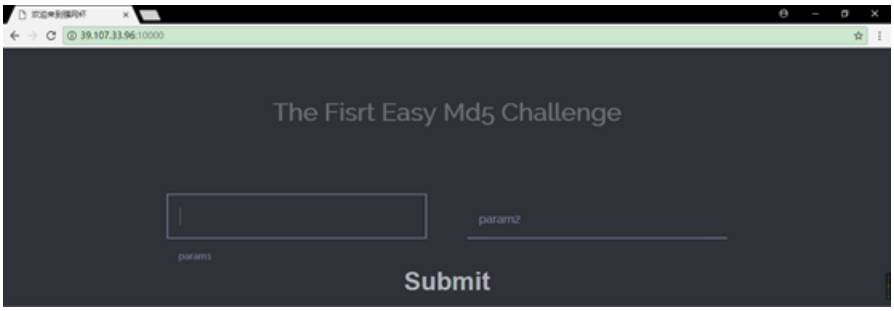
第二届强网杯部分writeup

转载

weixin_30325487 于 2018-03-24 23:15:00 发布 121 收藏
文章标签： php
原文链接：http://www.cnblogs.com/LLMF/p/8641992.html

MD5部分

第一题



一看就有些眼熟 emmmm

查看一下源代码：

重点是这里

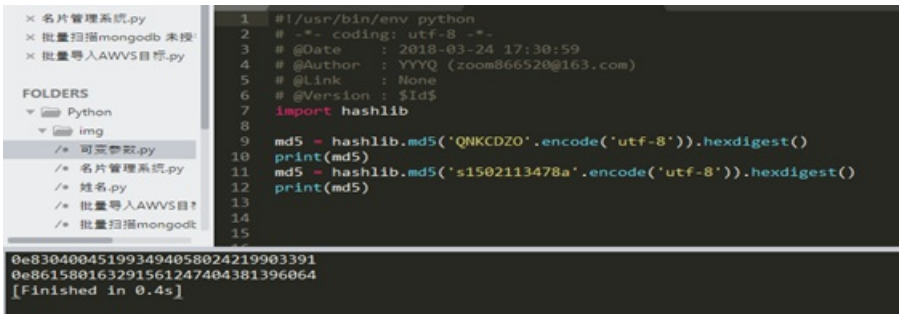
```
<!--
    if($_POST['param1']!=$_POST['param2'] && md5($_POST['param1'])==md5($_POST['param2'])){
            die("success!");
        }
```

这里面要求POST上去的参数 param1 != param2 && md5(‘param1’) == md5(‘param2’)

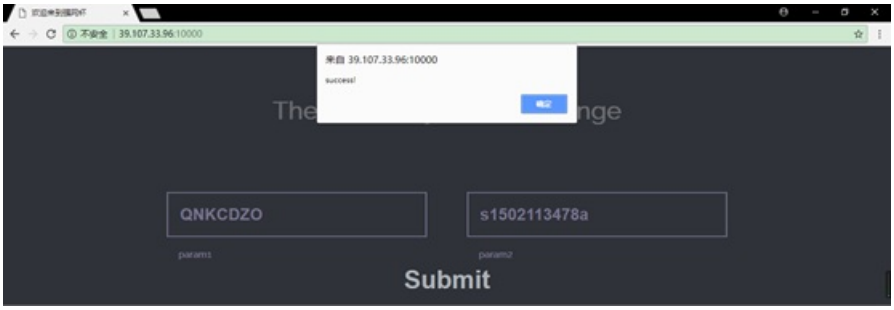
这道题运用了php的一个哈希比较缺陷，就是php在处理0e开头md5哈希字符串时，会将他看成 0 （具体下面那篇文章）PHP在处理哈希字符串时，会利用”!=”或”==”来对哈希值进行比较，它把每一个以”0E”开头的哈希值都解释为0，所以如果两个不同的密码经过哈希以后，其哈希值都是以”0E”开头的，那么PHP将会认为他们相同，都是0。

http://www.freebuf.com/news/67007.html

那么，我们构造两个不一样的字符，md5加密后以0E开头的，就可以验证成功

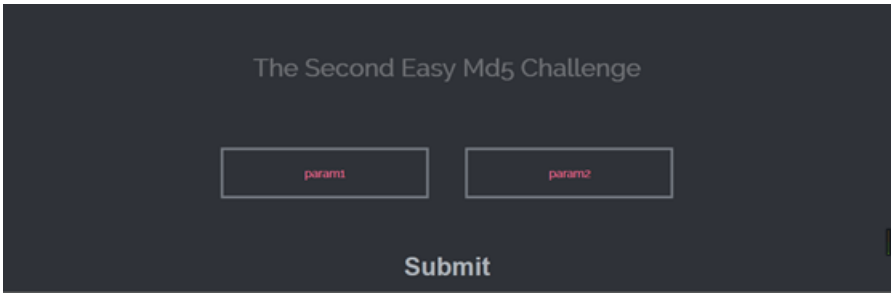


提交上去



成功

第二题



依旧是一道和MD5有关的

我们看一下源代码

此时比较符已经变成了 === 这样的类型 ， 我们可以知道，在php官方给出的补丁中，就是改用了===修复了这个缺陷。

在PHP中，我们可以知道，md5()这个函数是这样定义的：md5(string,raw)

也就是说，第一个必须是字符串，那么假设我们传入的不是一个字符串呢？

我们构造一下php代码看看

```php
<?php
$a=array("a");
$b=array();
var_dump(md5($a)==md5($b));
var_dump($a==$b);
?>
```

运行一下

```
Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 4

Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 4
bool(true) bool(false)
```

发现报错了

这里边我们构造了一个数组，传入到md5()这个函数里边，报错提示md5()第一个参数必须为str类型。但是程序依然会继续运行，我们修改一下代码看看是不是这样的

```php
<?php
$a=array("a");
$b=array();
var_dump(md5($a)==md5($b));
var_dump($a==$b);
if($a!==$b && md5($a)===md5($b))
    {
    die("success!");
    }
?>
```

这里我们加入判断，显然，if的条件是不符合的，那么，程序会不会输出seccess呢？

我们运行一下

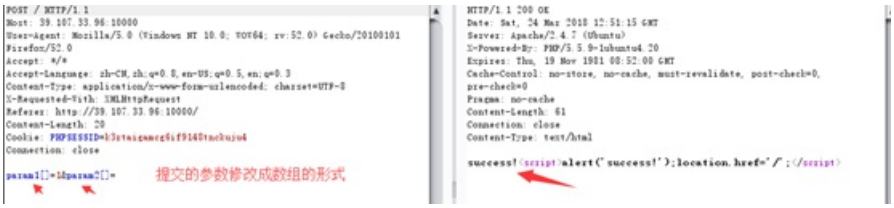
```
Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 4

Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 4
bool(true) bool(false)
Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 6

Warning: md5() expects parameter 1 to be string, array given in C:\phpStudy\PHPTutorial\WWW\test.php on line 6
success!
```

这里我们看到，即使条件不符合，也打印出了seccess

因此，这里我们只要通过burp抓包，将提交的参数改成数组，就可以了



第三题

先来看看源代码

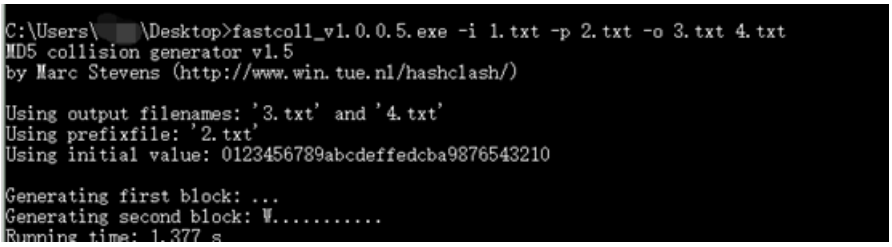
这里限定了str的类型，第二题的方法在这里就不适用了。

因此我们在这里，使用的是文件的碰撞。

我们用到了fastcoll_v1.0.0.5 这个软件，用来生成两个有着相同MD5值的文件。

先创建两个空的TXT文件，分别为1.txt和2.txt，两个文件名不一样即可

命令如下：



这边已经生成了两个文件，我们校验一下他们的MD5值

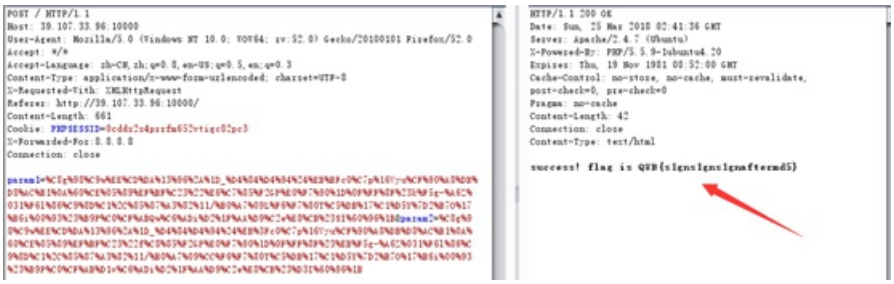
3.txt (128/128 b) | 耗时: 0.03s @ 3.67 KiB/s | 1521945981167	
CRC-32	841cc790
MD5 Hash	fce86dacf6bb4be401e1e56d342eb2da
SHA1 Hash	dcaf87f8a53a393fcdbf7d3e5e7e24b6976a99e2
SHA256 Hash	f20dfb2fce69533b4232eacf3709f3ac2eb16c458add9e05fd05bb9cc4ba2bb7
4.txt (128/128 b) | 耗时: 0.04s @ 3.04 KiB/s | 1521945981182	
CRC-32	88cab1ce
MD5 Hash	fce86dacf6bb4be401e1e56d342eb2da
SHA1 Hash	e8fbfc760a1f260c58d1334fb249e95a5853e8a0
SHA256 Hash	1d7eba6c0fb1d1ac04d5807e66a32ee465f6a4d0677f59106733e20fca796b3c

我们可以看到，他的MD5是相等的

那么我们怎么提交上去了？没错，就是URL编码成二进制

Py3代码如下

就得到了两个文件的编码，我们通过burp抓包，然后把我们的东西提交上去



成功的拿到了flag

PS：这里西安的大佬说要构造一下XFF头，虽然不知道为什么，但还是听了一下大佬的话。

自此，MD5部分完成。

转载于:https://www.cnblogs.com/LLMF/p/8641992.html