XCTF攻防世界_Web练习区

原创

FAFU小宋 于 2020-10-30 18:02:22 发布 218 收藏 3

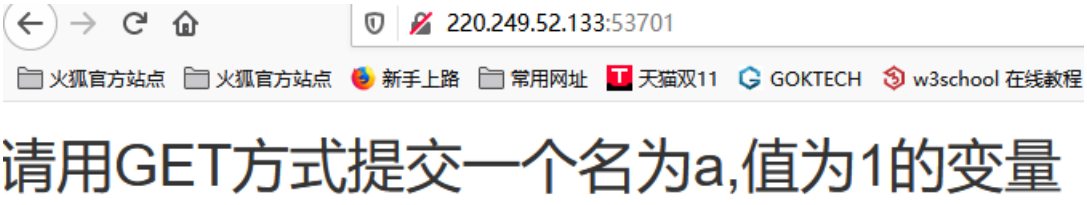
分类专栏： XCTF 文章标签： 安全

XCTF_Web_新手练习区

view_source

这里右键无法查看网页源代码，可通过f12键或者在url地址前加上"view-source:"查看网页源代码找到flag。

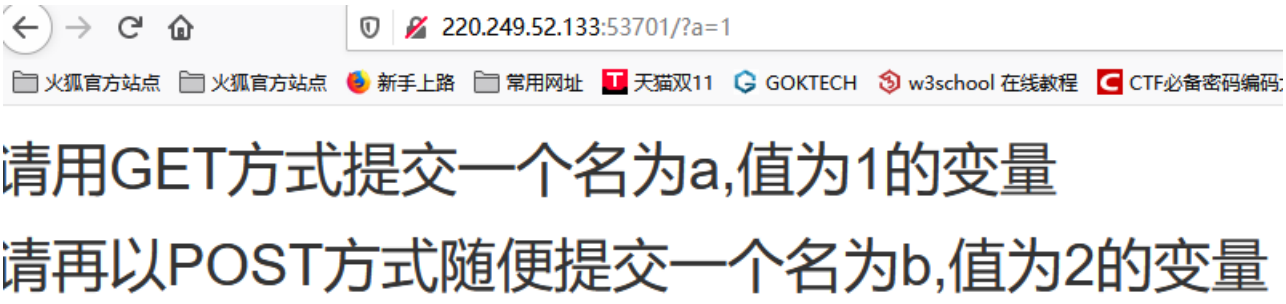
get_post



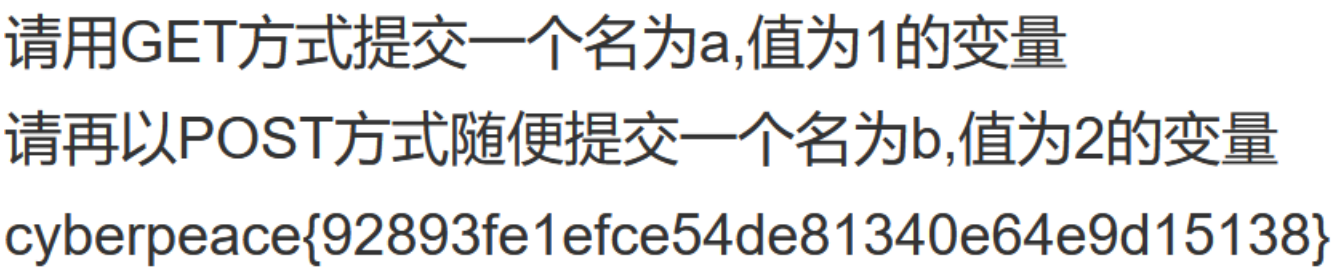
HTTP常用的请求方法：get，post。

（1）直接把请求参数拼接在URL后面，以？间隔URL和参数，若有多个参数，则以&间隔参数

```
http://220.249.52.133:55274/? a=1
```



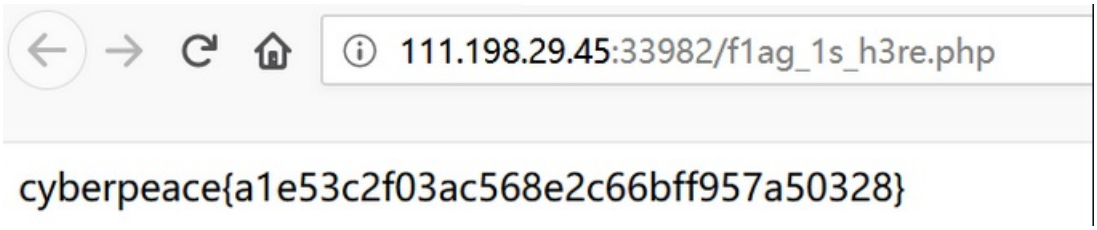
（2）post：提交post请求用插件hackbar。首先置入URL，选择POST Data，输入post请求，然后执行。



robots

（1）在URL后加上/robots.txt并访问，发现f1ag_1s_h3re.php

（2）访问http://111.198.29.45:33982/f1ag_1s_h3re.php得到flag



（3）也可使用扫目录脚本dirsearch(https://github.com/maurosoria/dirsearch)

```
python dirsearch.py -u http://10.10.10.175:32793/ -e *
```

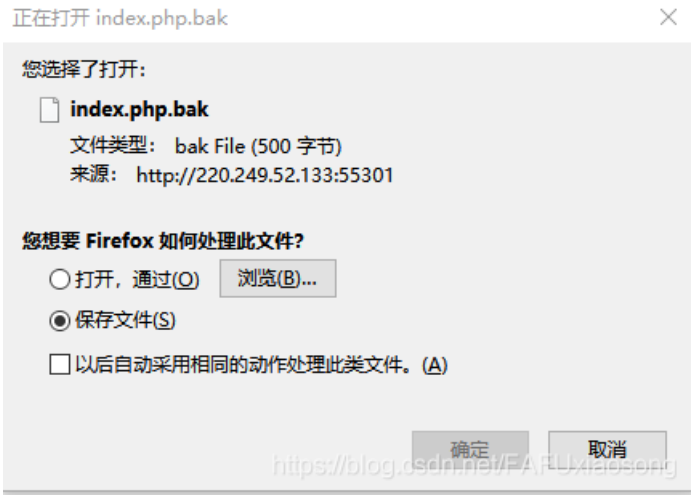
#使用dirsear需要在Python3环境下

#robots.txt是搜索引擎中访问网站的时候要查看的第一个文件。当一个搜索蜘蛛访问一个站点时，它会首先检查该站点根目录下是否存在robots.txt，如果存在，搜索机器人就会按照该文件中的内容来确定访问的范围；如果该文件不存在，所有的搜索蜘蛛将能够访问网站上所有没有被口令保护的页面。

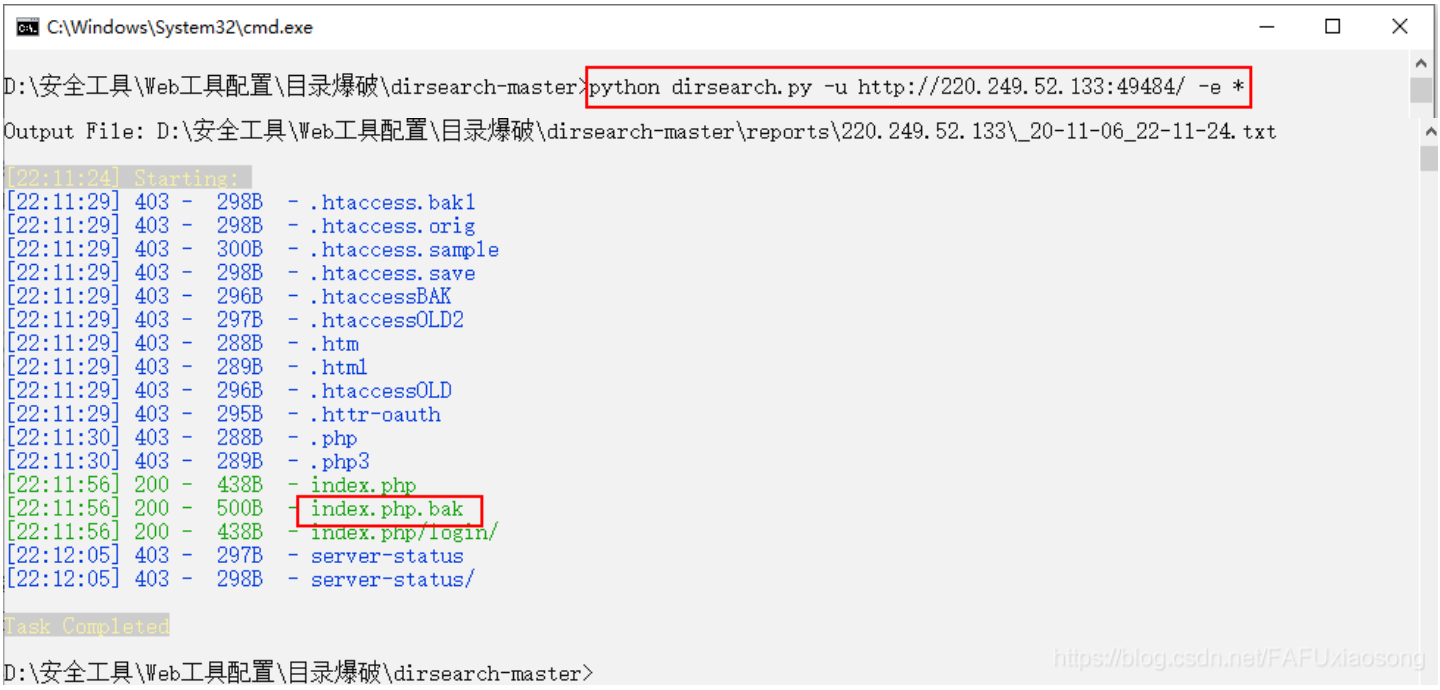
backup

你知道index.php的备份文件名吗?

（1）常见的备份文件后缀名有：".gif"、".svn"、".swp"、"~"、".bak"、".bash_history"、".bkf"（共7种）
根据提示index.php文件进行备份，逐个尝试构造备份链接。找到.bak的备份文件

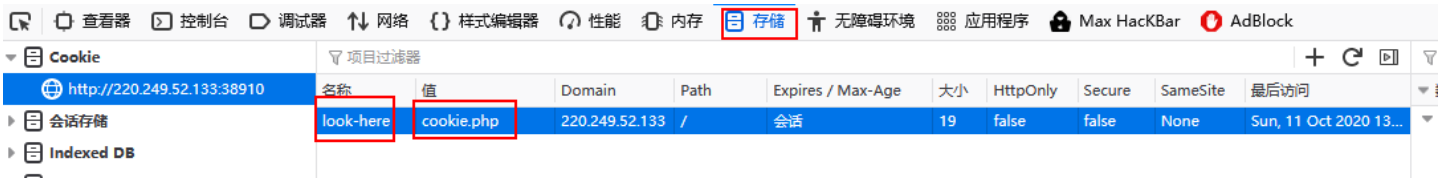


（2）下载备份文件直接打开就能够找到flag。
（3）也可以使用目录扫描工具dirsearch扫出index.php.bak文件，然后构造链接即可。

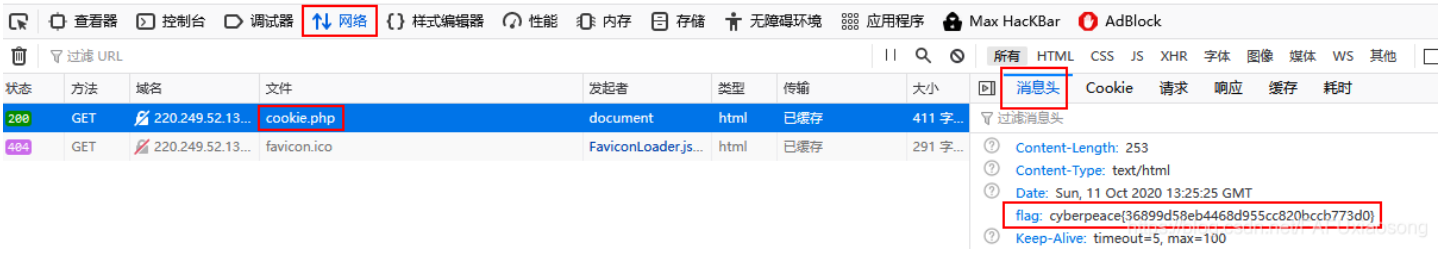


cookie

解法一：
（1）在Firefox浏览器按下F12键打开开发者工具，刷新后，在“存储”一栏，可看到名为look-here的cookie的值为cookie.php

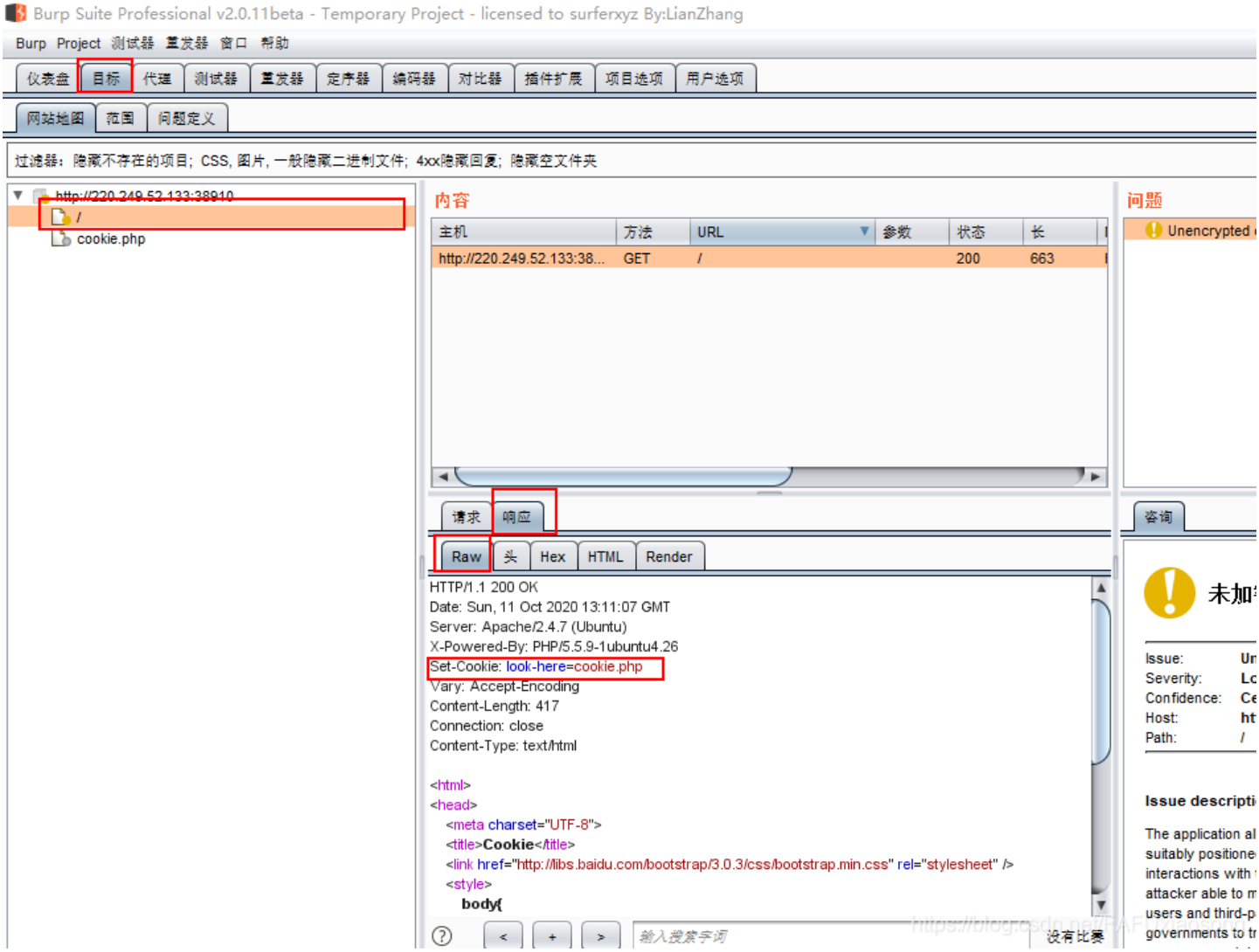


（2）构造链接访问http://220.249.52.133:38910/cookie.php，提示查看http响应包，在“网络”一栏，可看到访问cookie.php的数据包，在消息头内可发现flag

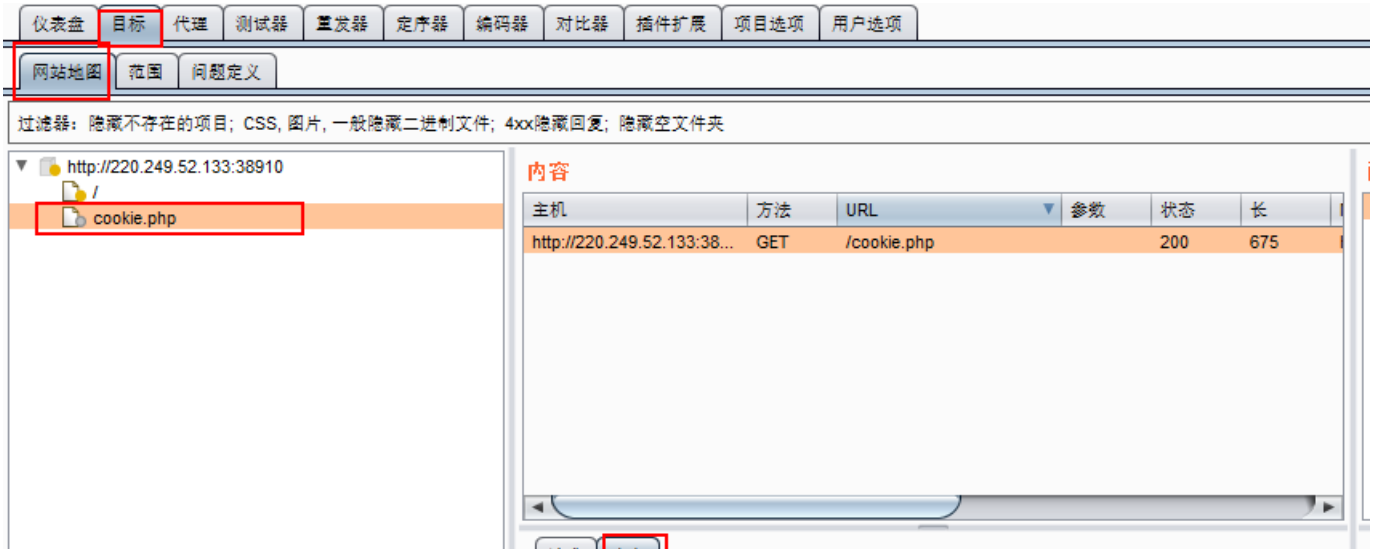


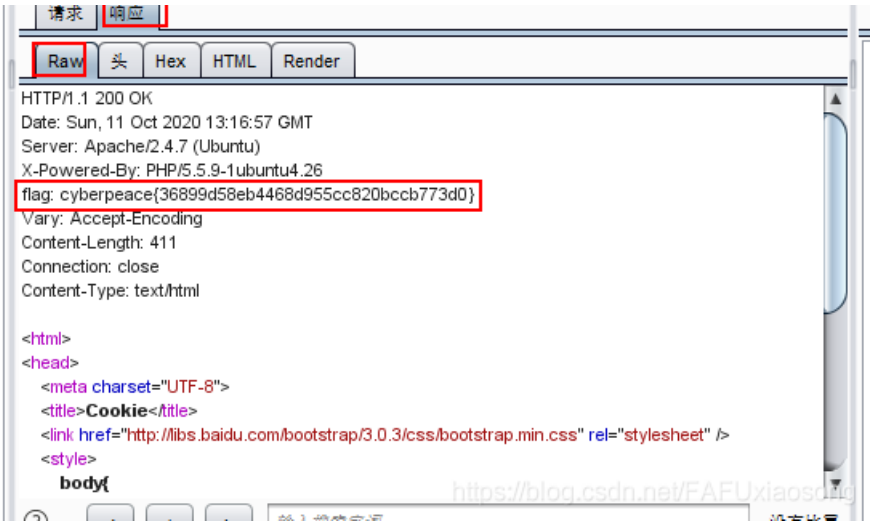
解法二：

（1）使用burpsuite工具扫描URL，然后查看response（响应），可看到名为look-here的cookie的值为cookie.php



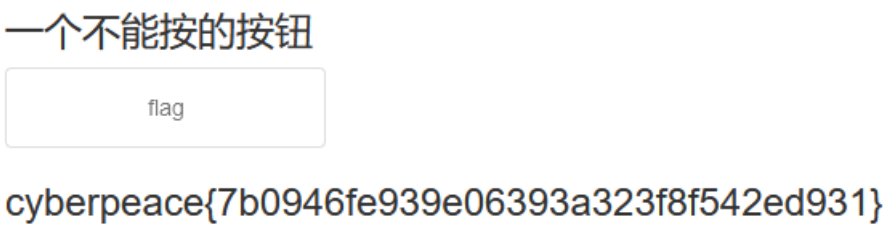
（2）继续扫描构造的链接http://220.249.52.133:38910/cookie.php，查看response获取flag





disabled_button

（1）使用Firefox浏览器按F12键打开开发者工具，在查看器窗口审查元素，发现存在disabled=""字段，在按钮中使用了该属性，右键编辑HTML，将该字段删除。按钮可按，即可得到flag。

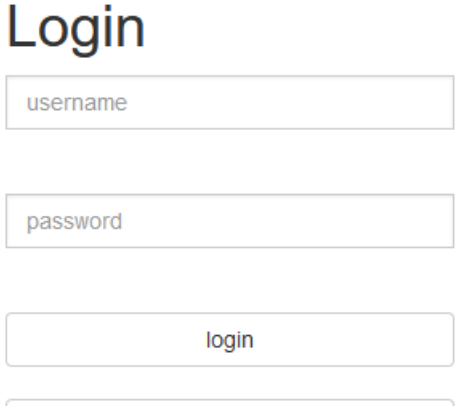


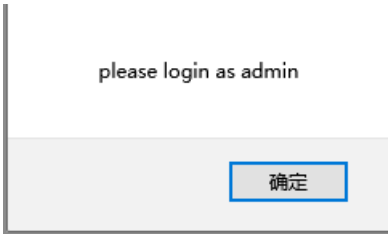
cyberpeace{7b0946fe939e06393a323f8f542ed931}

（2）使用Firefox的hackbar，用post方式传递auth=flag，同样可以得到flag。

weak_auth

（1）题目属于弱口令爆破，随便输入一个用户名和密码，提示要用admin用户登录，跳转到了check.php，在URL前加view-source:查看源代码，提示要用字典。

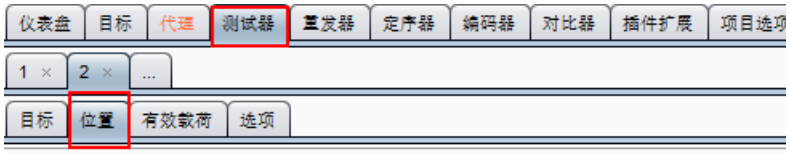




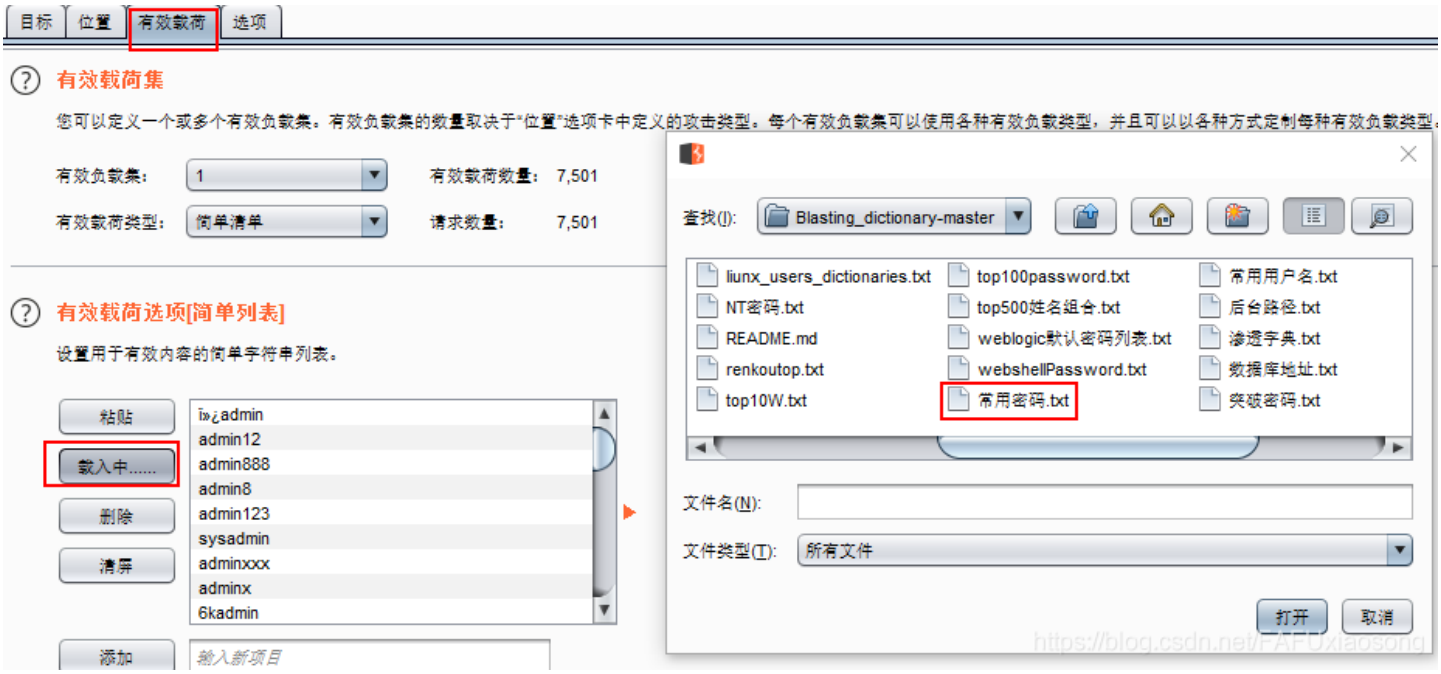

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>weak auth</title>
</head>
<body>

<!--maybe you need a dictionary-->


</body>
</html>

```


（2）在Firefox浏览器中“选项”–>"网络设置”–>“手动代理配置”开启foxproxy代理服务器，打开Burpsuite工具，点击代理—拦截请求，输入123密码再登录。

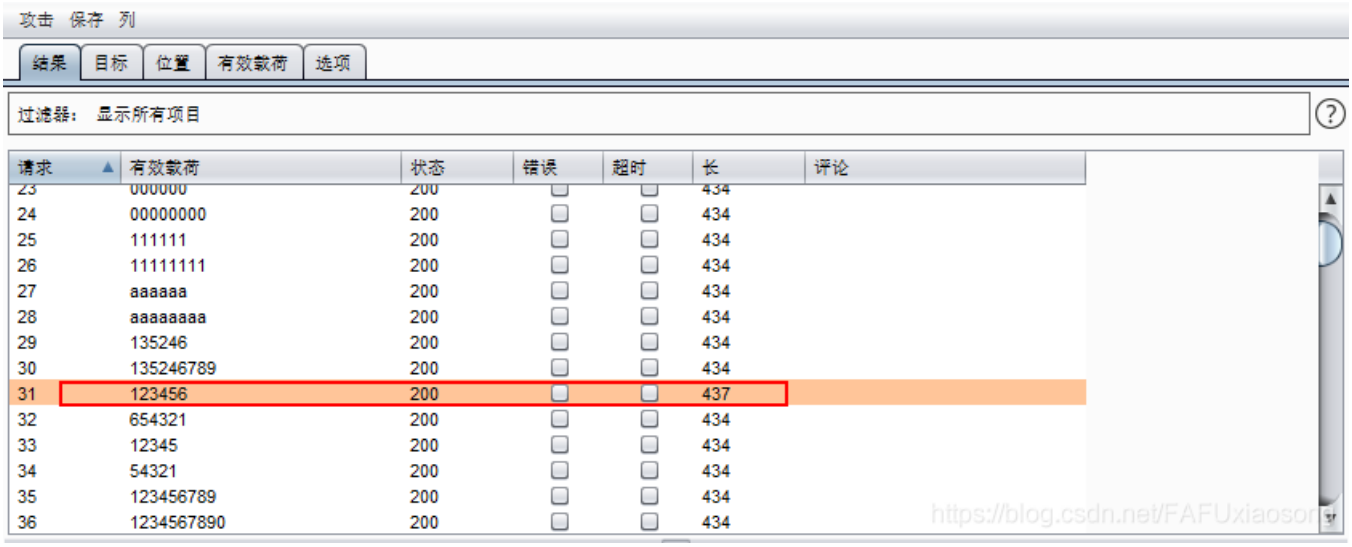
（3）设置爆破点：将123作为攻击对象，具体操作：测试器–>位置–>选中“123”–>添加



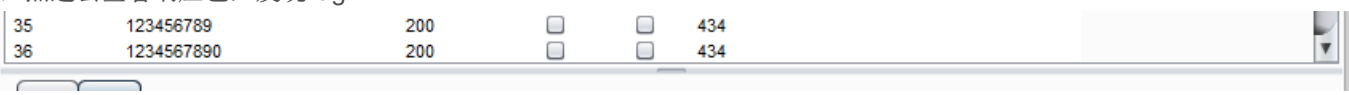
（5）加载字典：点击有效载荷–载入–常用密码.txt。字典:
<https://github.com/rootphantomer/Blasting_dictionary/blob/master/%E5%B8%B8%E7%94%A8%E5%AF%86%E7%A0%81.txt>

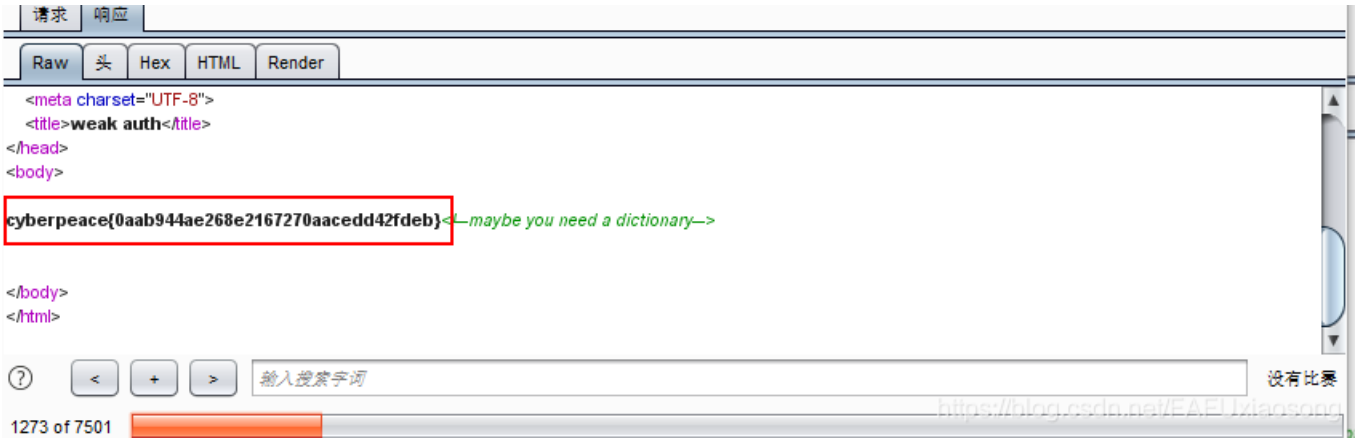


（6）开始攻击，查看响应包列表，发现密码为123456时，响应包的长度和别的不一样



（7）点进去查看响应包，发现flag




请求
响应
Raw
头
Hex
HTML
Render
<meta charset="UTF-8">
<title>weak auth</title>
</head>
<body>
cyberpeace{0aab944ae268e2167270aacedd42fdeb}<!--maybe you need a dictionary-->
</body>
</html>
输入搜索字词
没有比赛
1273 of 7501

simple_php

（1）打开页面，进行代码审计，发现需要输入a，b变量，同时满足a==0 且a为真，b不是数字且b>1234才回返回flag，用get方法输入变量，在URL后加“/?“即可开始赋值，传输多个参数则以”&"间隔。

```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```

（2）输入a=0无法得到flag1，因为不满足第二个a为真的条件，所以可以把参数a构造为’0’或abc这种转换后为0，但本身也为真的形式，即str类型。

```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```

Cyberpeace{647E37C7627CC3E401

（3）数字和字符混合的字符串转换为整数后只保留数字，所以b可以构造为12345a，类型转换后为12345，大于1234，得到flag。

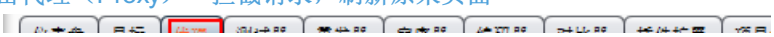
```php
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0  and  $a){
        echo  $flag1;
}
if(is_numeric($b)){
        exit();
}
if($b>1234){
        echo  $flag2;
}
?>
```

Cyberpeace{647E37C7627CC3E4019EC69324F66C7C}

xff_referer

（1）打开网页后，发现IP是123.123.123.123，使用Firefox浏览器，手动配置代理服务器

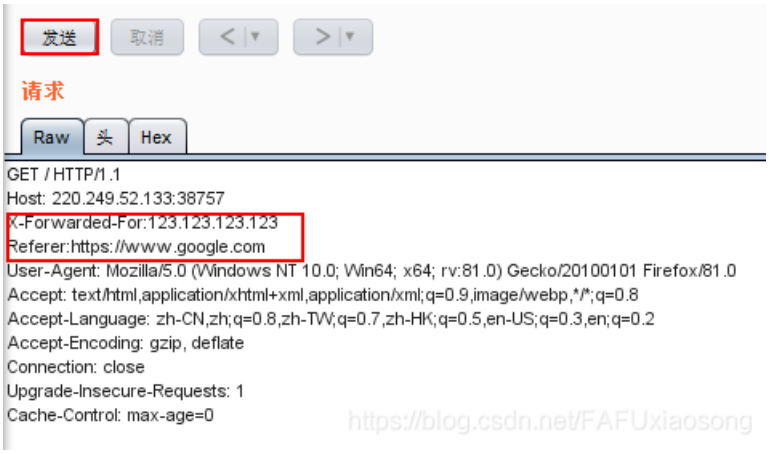
（2）打开Brup工具抓包，点击代理（Proxy）—拦截请求，刷新原来页面

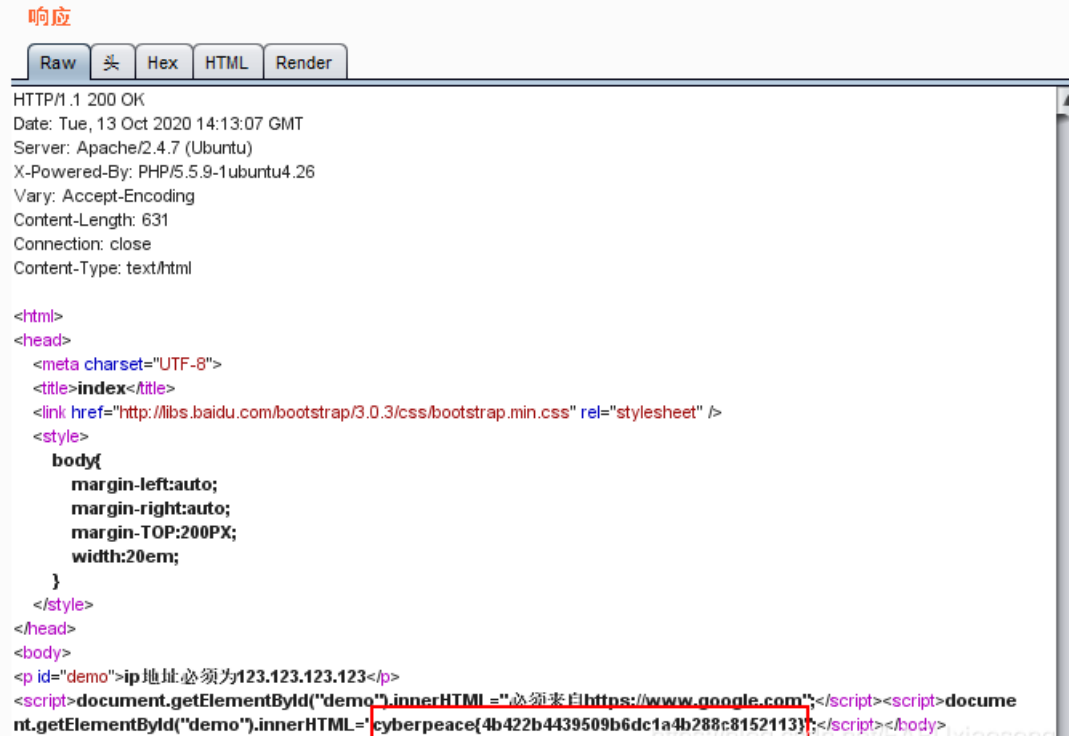


（3）右键选择发送到repeater。在repeater里查看目标地址内容，在Host下方添加：
X-Forwarded-For：123.123.123.123（这一步是伪造XFF，点击放包（Forward），收到提示）原来的页面变成了https://www.google.com

必须来自https://www.google.com

（4）在Repeater（重发器），Host下方添加：
X-Forwarded-For:123.123.123.123Referer:https://www.google.com (这一步是伪造Referer）点击发送，在响应中看到flag





</html>

#XFF漏洞攻击原理及防御方案：https://www.freebuf.com/company-information/220414.html

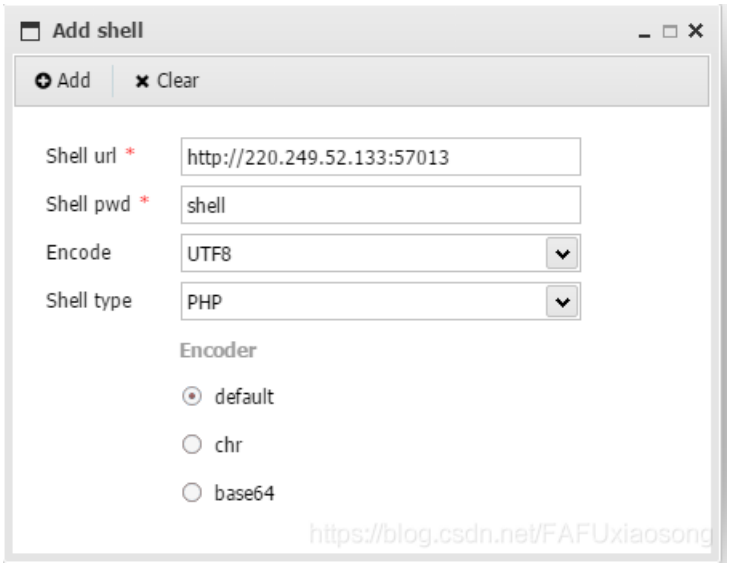
xff和referer:
#X-Forwarded-For：简称xff头，它代表客户端，也就是HTTP的请求端真实的IP，只有在通过了HTTP代理或者负载均衡服务器时才会添加该项。xff是http的拓展头部，作用是Web服务器获取访问用户的IP真实地址（可伪造）。由于很多用户通过代理服务器进行访问，服务器只能获取代理服务器的IP地址，而xff作用在于记录用户的真实IP，以及代理服务器的IP。格式为：
X-Forwarded-For：本机IP，代理1IP，代理2IP。

#HTTP Referer是header的一部分，当浏览器向Web服务器发送请求的时候，一般会带上Referer，告诉服务器我是从哪个页面链接过来的，服务器基于此可以获得一些有用的信息用于处理。Referer是http的拓展头部，作用是记录当前请求页面的来源页面的地址。服务器使用Referer确认访问来源，如果Referer内容不符合要求，服务器可以拦截或者重定向请求。

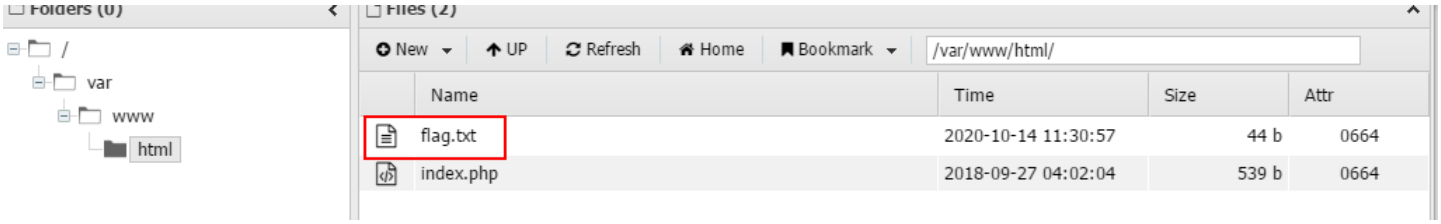
webshell

（1）一句话木马，直接提示给了php一句话，使用菜刀类工具连接（Cknife或中国蚁剑），口令就是shell

（2）这里使用中国蚁剑。右键–>Add,输入URL和口令，选择类型，点击"Add“即可



（3）右键–>“FileManager”文件管理，即可发现flag.txt文件，打开获取flag



command_execution

（1）尝试输入127.0.0.1，发现可以访问成功

```
ping -c 3 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.050 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.034 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.035 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.034/0.039/0.050/0.010 ms
```

（2）输入127.0.0.1 | ls（用&&也可以），看是否能够访问当前目录

```
ping -c 3 127.0.0.1 | ls
index.php
```

（3）访问成功，再试试寻找 .txt文件（或者直接寻找flag.txt文件也可），输入

```
127.0.0.1 | find / -name "*.txt"           //寻找.txt文件
127.0.0.1 | find / -name "flag.txt"        //寻找flag.txt文件
```

```
ping -c 3 127.0.0.1 | find / -name "flag.txt"
/home/flag.txt
```

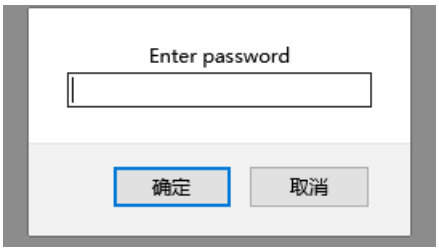
（4）访问成功后都可发现flag.txt文件，用127.0.0.1 | cat /home/flag.txt 即可查看到flag

```
ping -c 3 127.0.0.1 | cat /home/flag.txt
cyberpeace{9ca8da6d4963610593144de52a31fb25}
```

#命令拼接：管道符“|”，其功能为将前一个命令的结果传递给后一个命令作为输入
&&：前一条命令执行成功时，才执行后一条命令

```
command1 | command2                          //只输出2的结果
command1 && command2                        // 成功才执行2
```

simple_js



（1）打开页面需要输入密码，查看网页源代码，可以发现js代码

```
<html>
<head>
    <title>JS</title>
    <script type="text/javascript">
    function dechiffre(pass_enc){
        var pass = "70,65,85,88,32,80,65,83,83,87,79,82,68,32,72,65,72,65";
        var tab  = pass_enc.split(',');
                var tab2 = pass.split(',');var i,j,k,l=0,m,n,o,p = "";i = 0;j = tab.length;
                        k = j + (l) + (n=0);
                        n = tab2.length;
                        for(i = (o=0); i < (k = j = n); i++ ){o = tab[i-l];p += String.fromCharCode((o = tab2[i]));
                                if(i == 5)break;}
                        for(i = (o=0); i < (k = j = n); i++ ){
                        o = tab[i-l];
                                if(i > 5 && i < k-1)
                                        p += String.fromCharCode((o = tab2[i]));
                        }
        p += String.fromCharCode(tab2[17]);
        pass = p;return pass;
    }
    String["fromCharCode"](dechiffre("\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x
    h = window.prompt('Enter password');
    alert( dechiffre(h) );
```


（2）进行代码审计，发现不论输入什么都会跳到假密码，真密码位于 fromCharCode

（3）发现该字符串中用到C语言的转义字符\x，每个\x后面的两位是一个16进制数，直接用C语言printf输出即可得到对应的ASCII码值。

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    printf("\x35\x35\x2c\x35\x36\x2c\x35\x34\x2
    printf("\n");
```

```
D:\新建文件夹\未命名1.exe
55, 56, 54, 79, 115, 69, 114, 116, 107, 49, 50
```

（4）将得到的数字转为ASCII码值即可得到flag

```
int a[]={55,56, 54,79,115,69,114,116,107,49,50};
int i;
for(i=0;i<sizeof(a)/sizeof(a[0]);i++){
    printf("%c",(char)a[i]);
}
return 0;
```

```
786OsErtk12
```