

# bugku题解-web

原创

昏头小杨 于 2021-10-08 11:33:06 发布 148 收藏

分类专栏: [bugku题解-web](#) 文章标签: [html](#) [web service](#)

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/qq\\_51775369/article/details/120606884](https://blog.csdn.net/qq_51775369/article/details/120606884)

版权

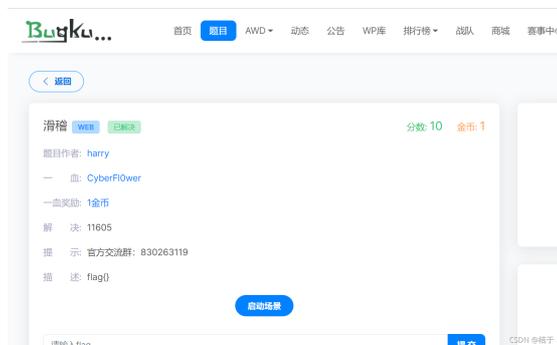


[bugku题解-web](#) 专栏收录该内容

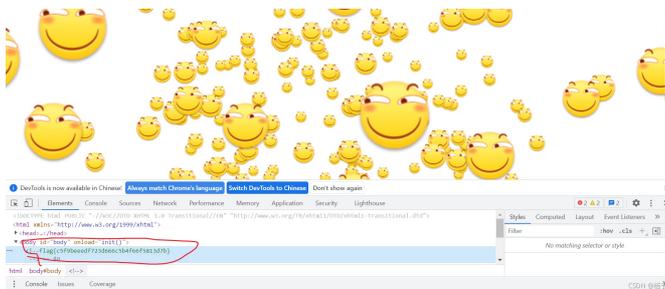
12 篇文章 1 订阅

订阅专栏

## 1.滑稽



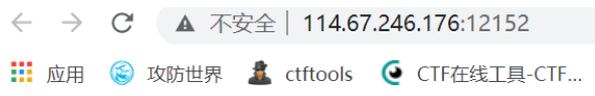
打开题目进去发现很多滑稽脸, 直接f12开发者, 就能看到flag



## 2. Simple\_SSTI\_1

打开题目

您需要传入一个名为 flag 的参数。



您需要传入一个名为 flag 的参数。

CSDN @桔子

f12查看源代码发现线索

```
</nead>...</nead>
<body>
  <font style="vertical-align: inherit;">...</font>
  <!-- You know, in the flask, we often set a secret_key variable.--> == $0
  <div id="goog-gt-tt" class="skiptranslate" dir="ltr">...</div>
  <div class="goog-te-spinner-pos">...</div>
</body>
</html>
```

CSDN @桔子

flag在secret\_key下，最下面说在flask里

- Flask/Jinja2
  - `{{ config.items() }}`
  - `{{ '.__class__.__mro__[-1].__subclasses__()' }}`

选择get的传入方式，一般secret-key中存在有价值的东西，或者说在config中，从而构建url: http://114.67.246.176:12152/?flag={{config}}, 看到secret\_key中存在着flag

```
<Config {'ENV': 'production', 'DEBUG': True, 'TESTING': False, 'PROPAGATE_EXCEPTIONS': None, 'PRESERV
'flag{a52a3fa4a97338797894cda1d02df410}', 'PERMANENT_SESSION_LIFETIME': datetime.timedelta(days
'APPLICATION_ROOT': '/', 'SESSION_COOKIE_NAME': 'session', 'SESSION_COOKIE_DOMAIN': False, 'SESSIO
True, 'SESSION_COOKIE_SECURE': False, 'SESSION_COOKIE_SAMESITE': None, 'SESSION_REFRESH_EACH_I
'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(seconds=43200), 'TRAP_BAD_REQUEST_ERRORS': N
'EXPLAIN_TEMPLATE_LOADING': False, 'PREFERRED_URL_SCHEME': 'http', 'JSON_AS_ASCII': True, 'JSON_S
'JSONIFY_MIMETYPE': 'application/json', 'TEMPLATES_AUTO_RELOAD': None, 'MAX_COOKIE_SIZE': 4093}>
```

CSDN @桔子

config的用法，config也是Flask模版中的一个全局对象，它包含了所有应用程序的配置值，所以可以使用config.xxx来查看该对象的属性值。

理解了题目的含义，直接构建如下url: http://114.67.246.176:12152/?flag={{config.SECRET\_KEY}} 得到flag

flag{a52a3fa4a97338797894cda1d02df410}

CSDN @桔子

python和jinja 2的的ssti模板注入题目，需了解基础的jinja语法和网页页面渲染方式

### 3.计算器

打开题目看到有一个算式，但是发现只能输入一位数

来源:B

CSDN @桔子

f12打开开发者工具

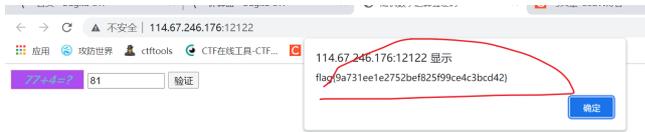
```
<span id="code" class="code" style="background: rgb(171, 77, 240); color: rgb(117, 166, 204);">77+4=?</sp
<input type="text" class="input" maxlength="1"
<button id="check">验证</button>
<div style="text-align:center;">...</div>
```

CSDN @桔子

发现长度是1，所以右键修改属性，改为2

```
e" class="code" style="background: rgb(171, 77, 240); color: white; padding: 2px;">text" class="input" maxlength="2" type="text" value="" == $0  
neck">验证</button>  
put  
CSDN @桔子
```

再次输入答案即可得到flag



CSDN @桔子

## 4.GET

Get是从服务器上获取数据，Get是把参数数据队列加到提交表单的Action属性所指的URL中，值和表单内各个字段一一对应，在URL中可以看到。

```
GET - Bugku CTF x 114.67.246.176:14394 x  
不安全 | 114.67.246.176:14394  
应用 攻防世界 ctftools CTF在线工具-CTF... csdn k  
$what=$_GET['what'];  
echo $what;  
if($what=='flag')  
echo 'flag{****}';
```

CSDN @桔子

根据提示，传入?what=flag即可



CSDN @桔子

## 5.POST

Post是向服务器传送数据。Post是通过HTTP post机制，将表单内各个字段与其内容放置在HTML header内一起传送到Action属性所指向的URL地址。用户看不到这个过程。

打开题目，用POST请求方法可知只要传递相应参数即可获得相应内容

```
POST - Bugku CTF x 114.67.246.176:13899 x  
不安全 | 114.67.246.176:13899  
应用 攻防世界 ctftools CTF在线工具-CTF... csdn k  
$what=$_POST['what'];  
echo $what;  
if($what=='flag')  
echo 'flag{****}';
```

CSDN @桔子

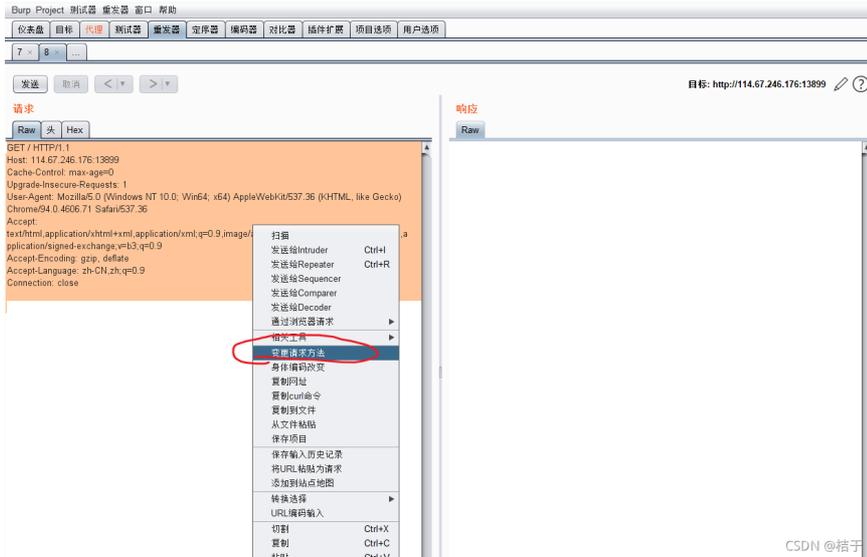
方法一：hackbar直接输入what=flag

方法二：bp抓包

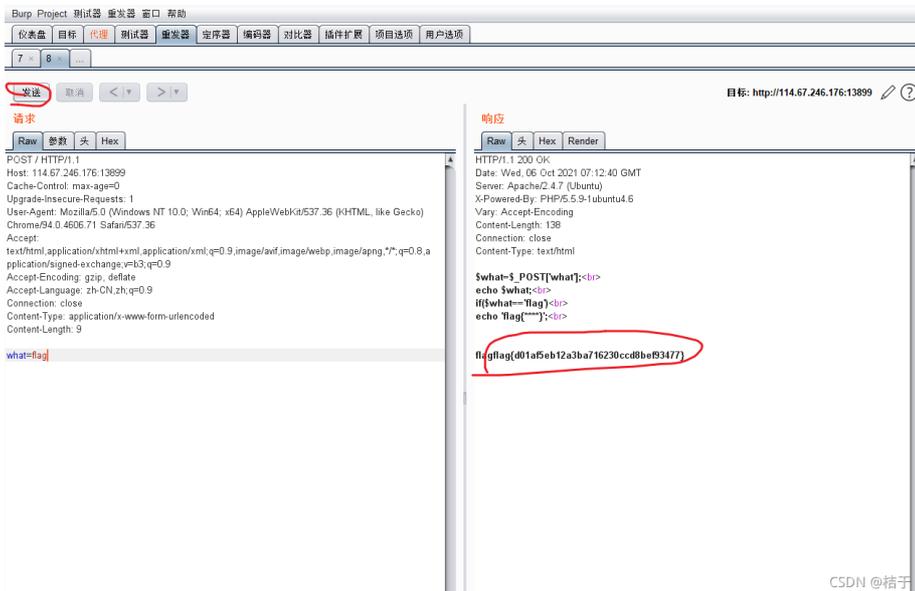


打开bp软件抓包

ctrl +r 发送，然后变更请求方式为post



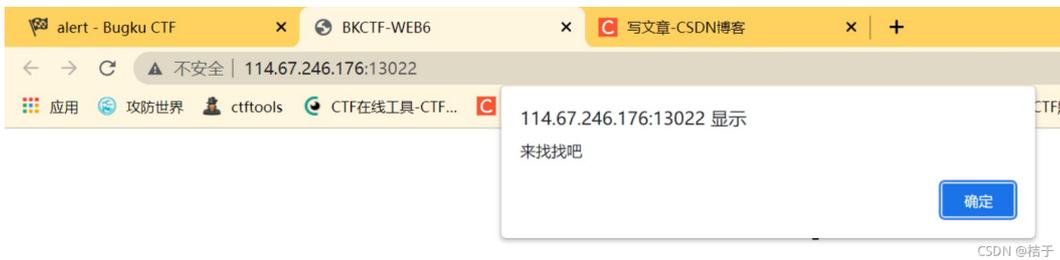
然后输入what=flag（与上面空一行）



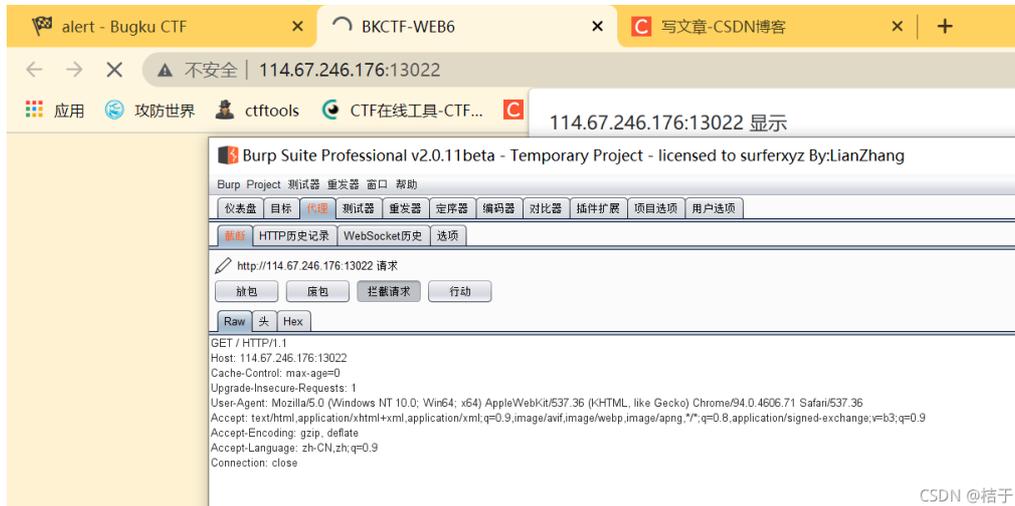
最后发送即可得到flag

## 6.alert

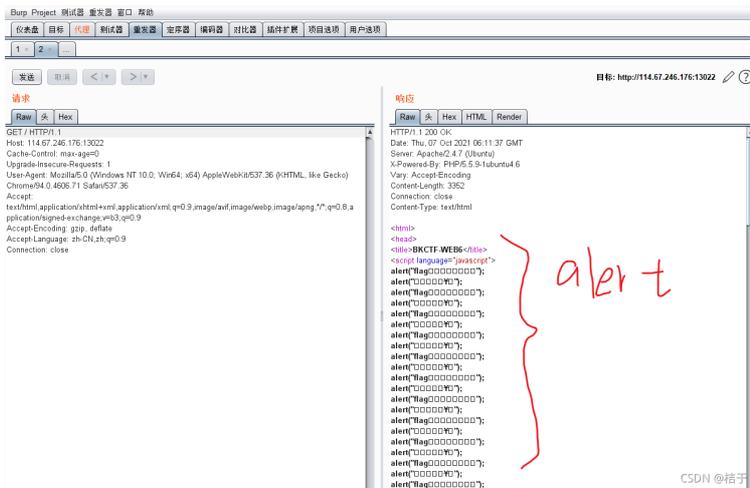
打开题目发现一直有弹窗



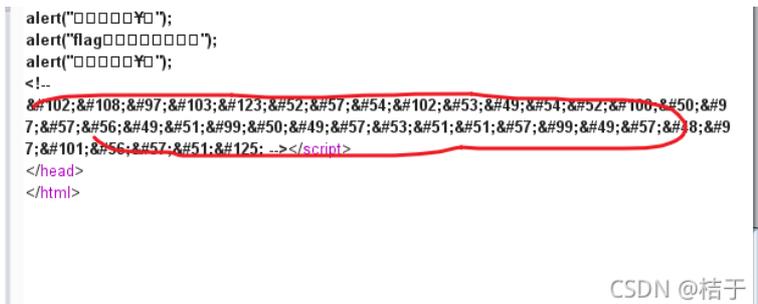
## 直接bp抓包



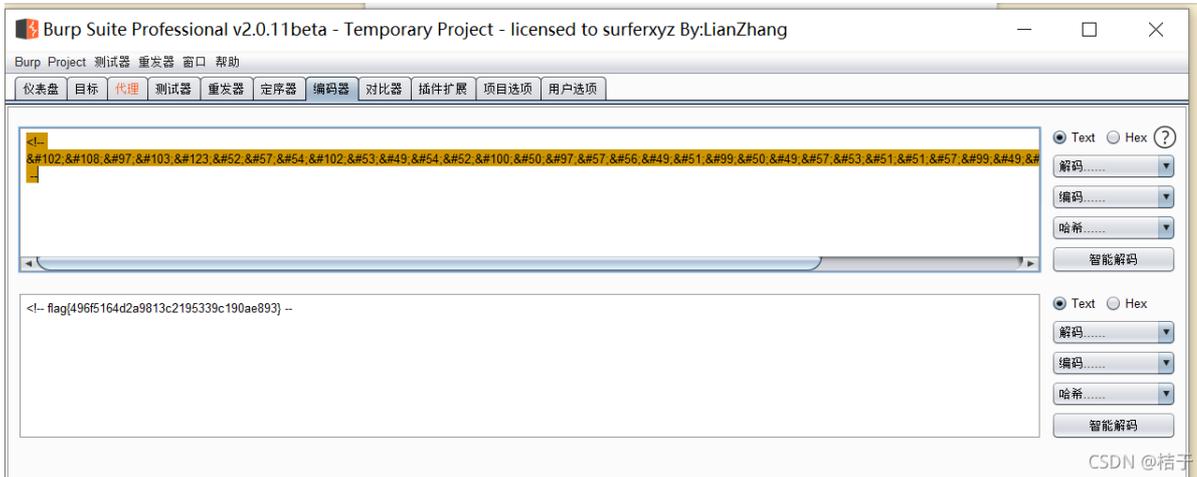
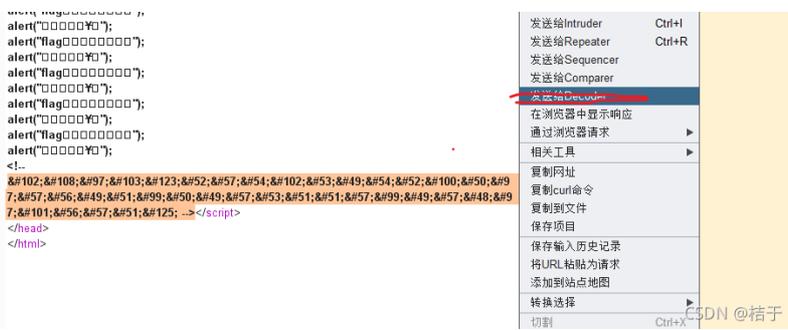
## 查看源码 (利用了 alert() 函数跳出弹窗)



## 发现最低端有一串字符

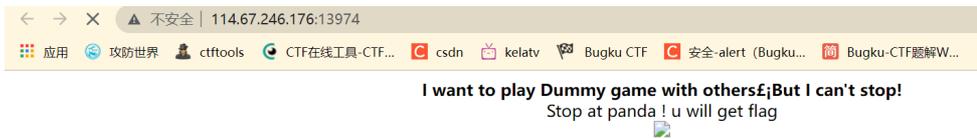


了解知识点: 末尾存在一些以 &# 开头 ; 号结尾的东西, 是html实体用burp解码得到flag

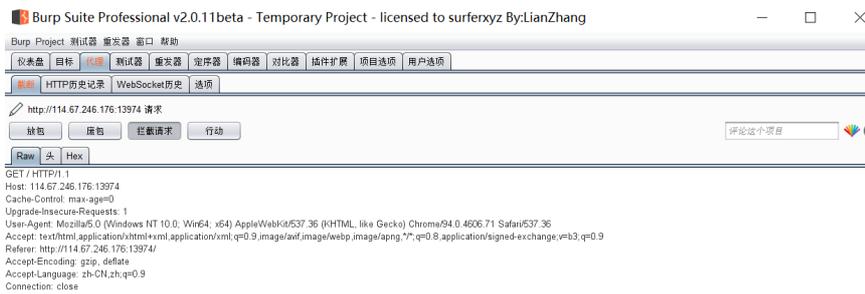


## 7.你必须让他停下

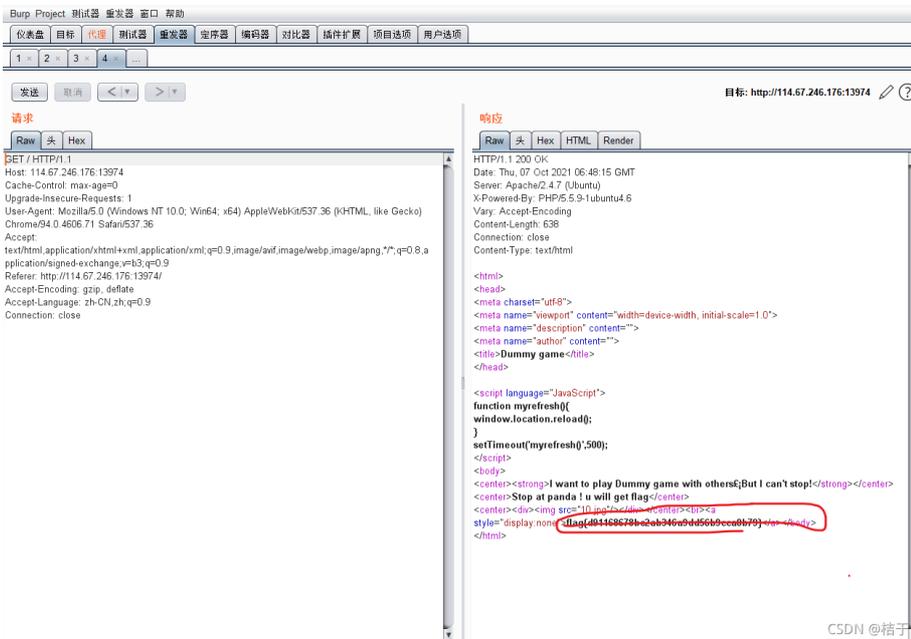
打开题目，发现页面一直刷新



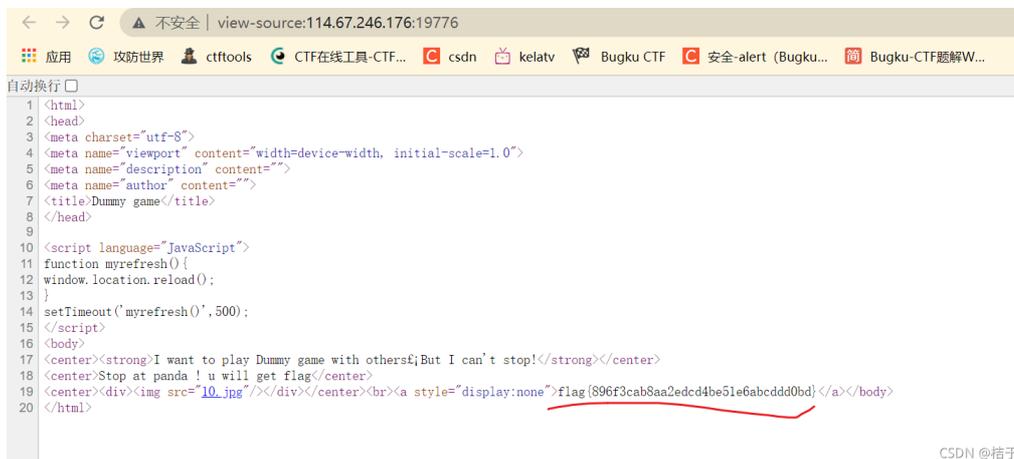
方法一：因为页面自动刷新，所以只需将burp收到的包发送repeater，查看响应，然后forward，如此反复，有一个响应有flag



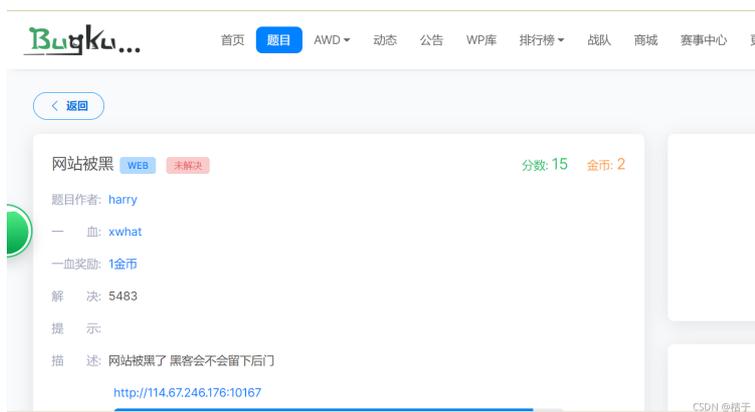
发送查看响应，然后forward，如此反复，有一个响应有flag



方法二：ctrl+u 查看源代码，一直刷新即可找到含有flag的页面源代码



## 8.网站被黑



打开题目进入一个网页



通过题目tips我们可以得知网站被黑之后，黑客留下了后门

使用御剑后台扫描工具 扫描到黑客留下的后门地址

《想念初恋》御剑后台扫描工具 珍藏版 By:御剑孤独 QQ:343034656

域名:  开始扫描 停止扫描

线程:  (条 CPU核心 \* 5最佳)  DIR: 446889  ASPX: 42529  探测200

超时:  (秒 超时的页面被丢弃)  ASP: 297812  PHP: 52815  探测403

MDB: 9071  JSP: 19739  探测3XX

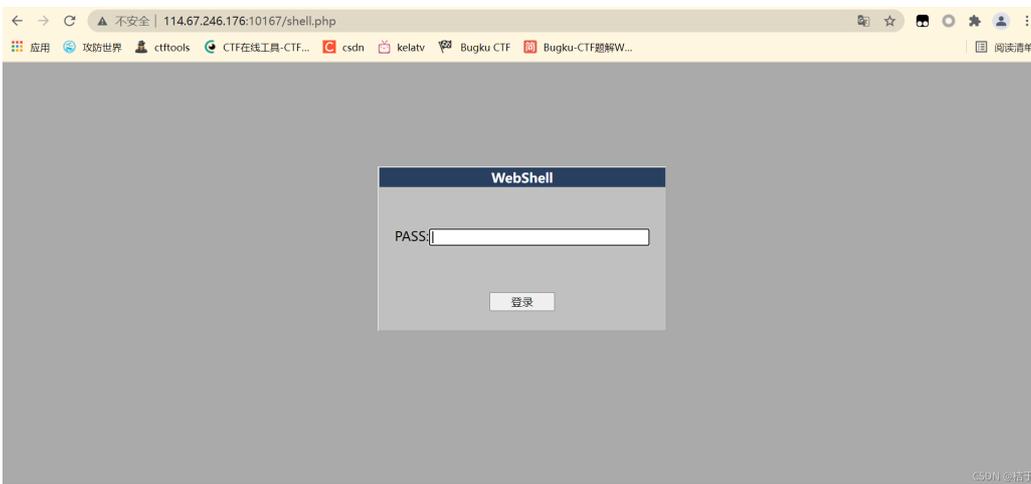
扫描信息: 扫描完成... 扫描线程: 0 扫描速度: 0/秒

ID	地址	HTTP响应
1	<a href="http://114.67.246.176:10167/index.php">http://114.67.246.176:10167/index.php</a>	200
2	<a href="http://114.67.246.176:10167/shell.php">http://114.67.246.176:10167/shell.php</a>	200
3	<a href="http://114.67.246.176:10167/index.php?.php">http://114.67.246.176:10167/index.php?.php</a>	200
4	<a href="http://114.67.246.176:10167/? .php">http://114.67.246.176:10167/? .php</a>	200

CSDN @桔子

通过扫描结果我们发现shell.php文件非常可疑

尝试访问，发现了一个需要密码的登录框



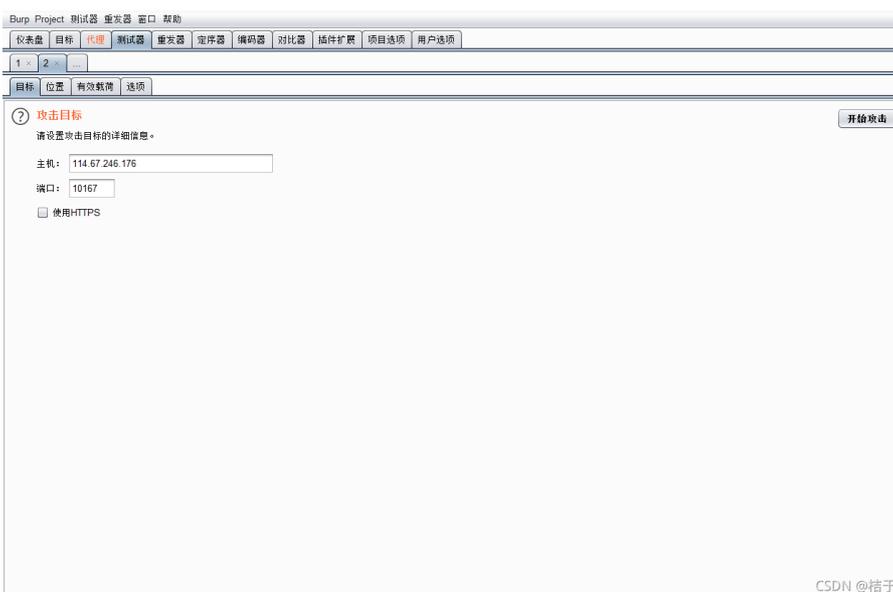
我们可以使用暴力破解的方式（我们使用Burpsuite进行抓包）



发送给intruder攻击（用burp暴力破解教程）[BurpSuite系列\(五\)---Intruder模块\(暴力破解\)\\_fendo-CSDN博客](#)

[BugKu Web题《网站被黑》writeUp\\_CSDN\\_suuny的博客-CSDN博客](#)

上面文章有爆破教程，学习后设置好开始爆破



得到密码为**hack**，输入得到**flag**

攻击 保存 列

结果 目标 位置 有效载荷 选项

过滤器: 显示所有项目

请求	有效载荷	状态	错误	超时	长	评论
173	hack	200	<input type="checkbox"/>	<input type="checkbox"/>	1203	
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
1	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
2	admin12	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
3	admin888	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
4	admin8	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
5	admin123	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
6	sysadmin	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
7	adminxxx	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
9	6kadmin	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	
8	adminx	200	<input type="checkbox"/>	<input type="checkbox"/>	1200	

CSDN @桔子



WebShell

PASS:

登录

flag{dae72141e3587c71580e610d7c61b886}

CSDN @桔子

## 9.本地管理员

< 返回

本地管理员 WEB 未解决 分数: 15 金币: 2

题目作者: harry

— 血: dotast

—血奖励: 1金币

解 决: 5006

提 示:

描 述: 本地管理员

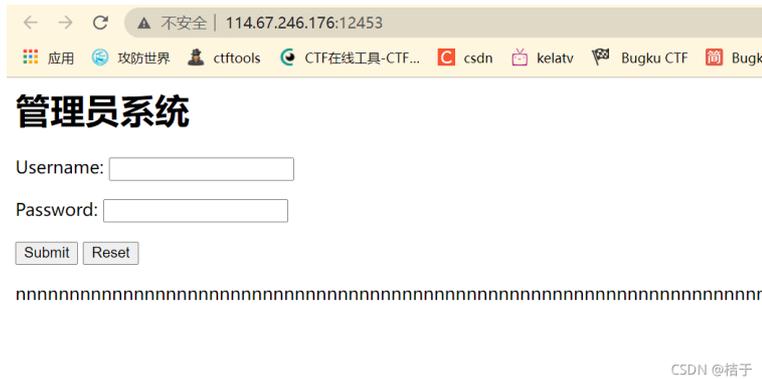
<http://114.67.246.176:12453>

02:42:06

删除场景 延时场景

CSDN @桔子

打开题目

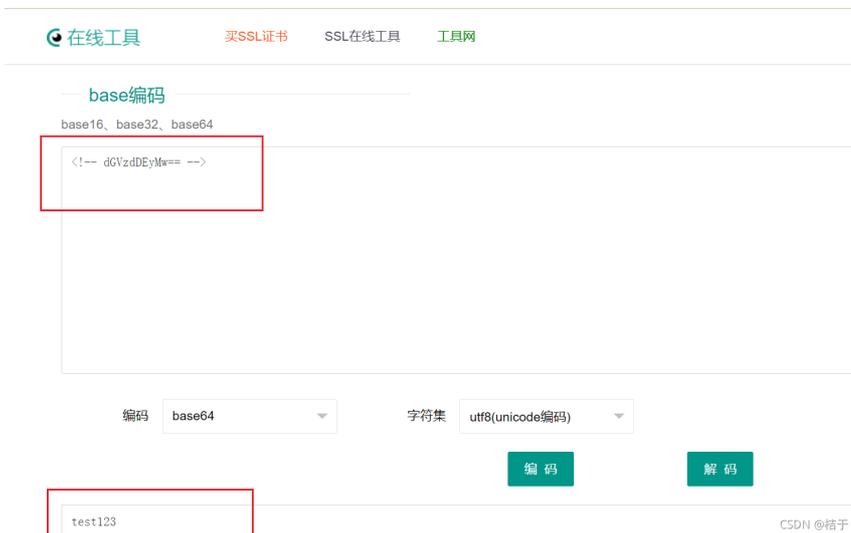


尝试登陆：admin和admin账号密码，显示IP禁止访问，IP已经被记录，打开控制台，发现底部有个`



CSDN @桔子

很明显的base64编码，解码得到：test123，猜测账号：admin，密码：test123



CSDN @桔子

应该就不是爆用户名和密码的问题了

又题目描述本地管理员，联系本地管理员登陆，burp抓包伪造XFF头

[IP欺骗\(XFF头等\)\\_weixin\\_30466421的博客-CSDN博客](#)（知识点）

### 1.X-Forwarded-For:（转载上面链接）

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。它不是RFC中定义的标准请求头信息，在squid缓存代理服务器开发文档中可以找到该项的详细介绍。标准格式如下：X-Forwarded-For: client1, proxy1, proxy2。

这一HTTP头一般格式如下：

X-Forwarded-For: client1, proxy1, proxy2, proxy3

其中的值通过一个 逗号+空格 把多个IP地址区分开, 最左边(client1)是最原始客户端的IP地址, 代理服务器每成功收到一个请求, 就把请求来源IP地址添加到右边。在上面这个例子中, 这个请求成功通过了三台代理服务器: proxy1, proxy2 及 proxy3。请求由client1发出, 到达了proxy3(proxy3可能是请求的终点)。请求刚从client1中发出时, XFF是空的, 请求被发往proxy1; 通过proxy1的时候, client1被添加到XFF中, 之后请求被发往proxy2; 通过proxy2的时候, proxy1被添加到XFF中, 之后请求被发往proxy3; 通过proxy3时, proxy2被添加到XFF中, 之后请求的去向不明, 如果proxy3不是请求终点, 请求会被继续转发。

如果一个 HTTP 请求到达服务器之前, 经过了三个代理 Proxy1、Proxy2、Proxy3, IP 分别为 IP1、IP2、IP3, 用户真实 IP 为 IP0, 那么按照 XFF 标准, 服务端最终会收到这样信息: X-Forwarded-For: IP0, IP1, IP2

鉴于伪造这一字段非常容易, 应该谨慎使用X-Forwarded-For字段。正常情况下XFF中最后一个IP地址是最后一个代理服务器的IP地址, 这通常是一个比较可靠的信息来源。burp抓包伪造XFF头

## 直接burp抓包伪造XFF头



伪造

## X-Forwarded-For 127.0.0.1 (空格再输入127.0.01)

## 得到flag

