

# XCTF\_Web\_新手练习区: get\_post

原创

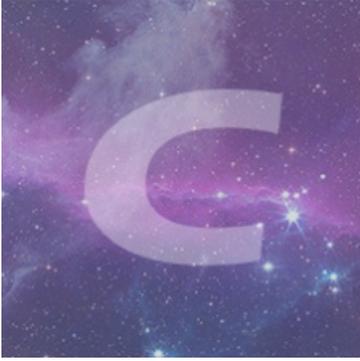
1stPeak 于 2020-02-16 10:43:40 发布 412 收藏 3

分类专栏: [CTF刷题](#) 文章标签: [XCTF](#)

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\\_41617034/article/details/90646126](https://blog.csdn.net/qq_41617034/article/details/90646126)

版权



[CTF刷题](#) 专栏收录该内容

87 篇文章 22 订阅

订阅专栏

## 第二题: get\_post

get post 查看Writeup 题目建议

难度系数: ★ 1.0

题目来源: Cyberpeace-n3k0

题目描述: X老师告诉小宁同学HTTP通常使用两种请求方法, 你知道是哪两种吗?

题目场景: 点击获取在线场景

题目附件: 暂无

[https://blog.csdn.net/qq\\_41617034](https://blog.csdn.net/qq_41617034)

### 目标:

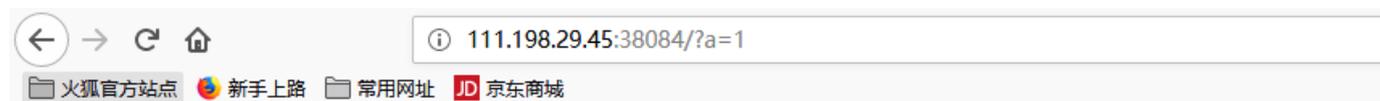
了解http请求方法, 此处考察get和post两个最常用的请求方法。

HTTP协议中共定义了八种方法或者叫“动作”来表明对Request-URI指定的资源的不同操作方式, 具体介绍如下:

- GET: 向特定的资源发出请求。
- POST: 向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的创建和/或已有资源的修改。
- OPTIONS: 返回服务器针对特定资源所支持的HTTP请求方法。也可以利用向Web服务器发送'\*'的请求来测试服务器的功能性。
- HEAD: 向服务器索要GET请求相一致的响应，只不过响应体将不会被返回。这一方法可以在不必传输整个响应内容的情况下，就可以获取包含在响应消息头中的元信息。
- PUT: 向指定资源位置上传其最新内容。
- DELETE: 请求服务器删除Request-URI所标识的资源。
- TRACE: 回显服务器收到的请求，主要用于测试或诊断。
- CONNECT: HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

## Writeup

打开指定url，按要求进行解决，这里共有两种方法

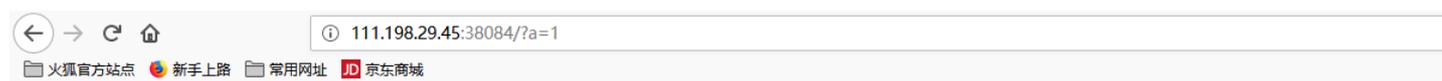


请用GET方式提交一个名为a,值为1的变量

请再以POST方式随便提交一个名为b,值为2的变量

[https://blog.csdn.net/qq\\_41617034](https://blog.csdn.net/qq_41617034)

方法一:



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{cf070d1a4ba246b1a9e74a3ae41f3f5a}



方法二:

## 使用Burpsuite

The screenshot shows the Burp Suite interface with a 'Request' tab on the left and a 'Response' tab on the right. The 'Request' tab shows a GET request to `/?a=1` with various headers including `Host`, `User-Agent`, `Accept`, `Accept-Language`, `Accept-Encoding`, `Connection`, and `Upgrade-Insecure-Requests`. The 'Response' tab shows an HTML response with headers like `Date`, `Server`, `X-Powered-By`, `Vary`, `Content-Length`, `Connection`, and `Content-Type`. The body of the response is HTML code that includes a title `<title>POST&GET</title>` and two `<h1>` tags with instructions in Chinese.

```
Request
Raw Params Headers Hex
GET /?a=1 HTTP/1.1
Host: 111.198.29.45:38084
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:67.0) Gecko/20100101 Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;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
Connection: close
Upgrade-Insecure-Requests: 1

Response
Raw Headers Hex HTML Render
HTTP/1.1 200 OK
Date: Tue, 28 May 2019 11:36:04 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.26
Vary: Accept-Encoding
Content-Length: 374
Connection: close
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>POST&GET</title>
  <link href="http://libs.baidu.com/bootstrap/3.0.3/css/bootstrap.min.css" rel="stylesheet" />
</head>
<body>

<h1>请用GET方式提交一个名为a,值为1的变量</h1>

<h1>请再以POST方式随便提交一个名为b,值为2的变量</h1>
</body>
</html>
```

这里的POST请求如何请求呢？

将 `GET /?a=1 HTTP/1.1` 修改为 `POST /?a=1 HTTP/1.1`

在最后一行添加 `Content-Type: application/x-www-form-urlencoded`

在最后一行下的第二行输入post的参数与其值（固定格式）

The screenshot shows the Burp Suite interface with a 'Request' tab on the left and a 'Response' tab on the right. The 'Request' tab shows a POST request to `/?a=1` with headers including `Host`, `User-Agent`, `Accept`, `Accept-Language`, `Accept-Encoding`, `Connection`, `Upgrade-Insecure-Requests`, `Content-Type`, and `Content-Length`. The body of the request is `b=2`. The 'Response' tab shows an HTML response with headers like `Date`, `Server`, `X-Powered-By`, `Vary`, `Content-Length`, `Connection`, and `Content-Type`. The body of the response is HTML code that includes a title `<title>POST&GET</title>` and two `<h1>` tags with instructions in Chinese, including a long alphanumeric string.

```
Request
Raw Params Headers Hex
POST /?a=1 HTTP/1.1
Host: 111.198.29.45:38084
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:67.0) Gecko/20100101 Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;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
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 5
b=2

Response
Raw Headers Hex HTML Render
HTTP/1.1 200 OK
Date: Tue, 28 May 2019 11:44:11 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.26
Vary: Accept-Encoding
Content-Length: 427
Connection: close
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>POST&GET</title>
  <link href="http://libs.baidu.com/bootstrap/3.0.3/css/bootstrap.min.css" rel="stylesheet" />
</head>
<body>

<h1>请用GET方式提交一个名为a,值为1的变量</h1>

<h1>请再以POST方式随便提交一个名为b,值为2的变量</h1><h1>cyberpeace{cf070d1a4ba246b1a9e74a3ae41f3f5a}</h1>
</body>
</html>
```

**注：**

http header里的Content-Type一般有这三种：

**application/x-www-form-urlencoded**：数据被编码为名称/值对。这是标准的编码格式。

**multipart/form-data**：数据被编码为一条消息，页上的每个控件对应消息中的一个部分。

**text/plain**：数据以纯文本形式(text/json/xml/html)进行编码，其中不含任何控件或格式字符。postman软件里标的是RAW。

form的enctype属性为编码方式，常用有两种：`application/x-www-form-urlencoded`和`multipart/form-data`，默认为`application/x-www-form-urlencoded`。

当action为get时候，浏览器用x-www-form-urlencoded的编码方式把form数据转换成一个字串

(name1=value1&name2=value2...)，然后把这个字串追加到url后面，用?分割，加载这个新的url。（这个了解就好）

当action为post时候，浏览器把form数据封装到http body中，然后发送到server。如果没有type=file的控件，用默认的application/x-www-form-urlencoded就可以了。但是如果有type=file的话，就要用到multipart/form-data了。（这里主要是用到这个）