

THM-HTTP in detail(HTTP协议介绍)-学习

本文相关的TryHackMe实验房间链接：<https://tryhackme.com/room/httpindetail>

通过学习相关知识点：了解如何使用 HTTP 协议向 Web 服务器请求内容。

H2 什么是 HTTP(S)?

什么是 HTTP? (HyperText Transfer Protocol-超文本传输协议)

HTTP 是你浏览网站时使用的，由 Tim Berners-Lee 和他的团队在 1989-1991 年间开发。HTTP 是用于与 web 服务器进行通信以便传输网页数据的一组规则，这些网页数据包括 HTML、图像、视频等。

什么是 HTTPS? (HyperText Transfer Protocol Secure)

HTTPS 是 HTTP 的安全版本。HTTPS 数据经过了加密处理，因此它不仅可以阻止第三方看到你正在接收和发送的数据内容，而且还可以确保你正在与正确的 Web 服务器进行通信（而不是其他冒充你所访问的目标 web 服务器的东西）。

答题

回答以下问题

HTTP 代表什么？

HyperText Transfer Protocol

正确答案

HTTPS 中的 S 代表什么？

secure

正确答案

右边的模拟网页有一个问题，找到后点击它。什么是挑战旗？

THM{INVALID_HTTP_CERT}

正确答案

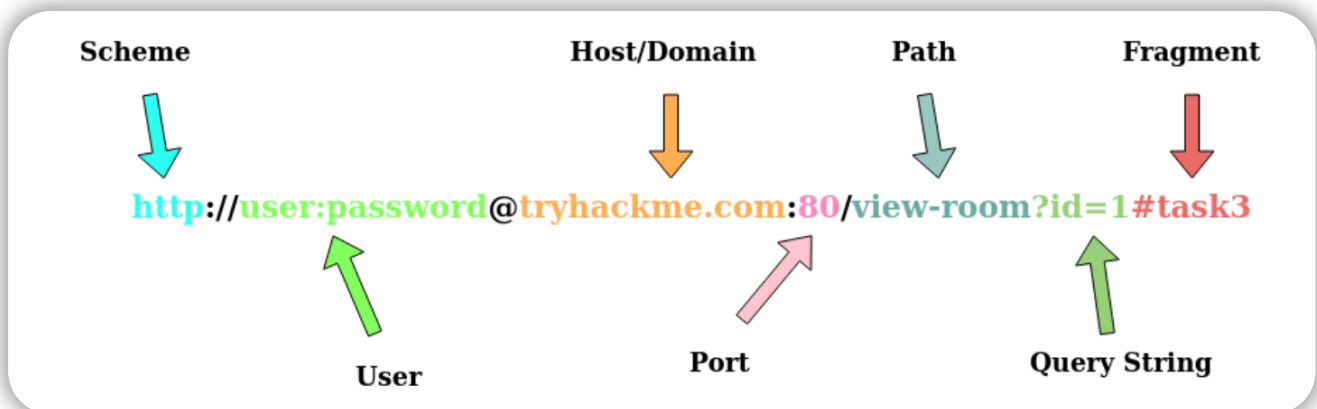


H2 请求和响应

当我们访问一个网站时，你的浏览器将需要向一个web服务器发出请求，以获得诸如 HTML、图像等资源，并根据来自web服务器的响应得到下载权限。在此之前，你需要明确地告诉浏览器如何获取以及在哪儿访问具体的资源文件，这里就需要 URL 来提供一些帮助。

什么是 URL? (Uniform Resource Locator)

如果你使用过互联网，那么你就使用过 URL，URL 主要用于指示如何访问互联网上的资源。下面的图片显示了一个 URL 的所有特性(它可能不会在每个请求中使用所有特性)。

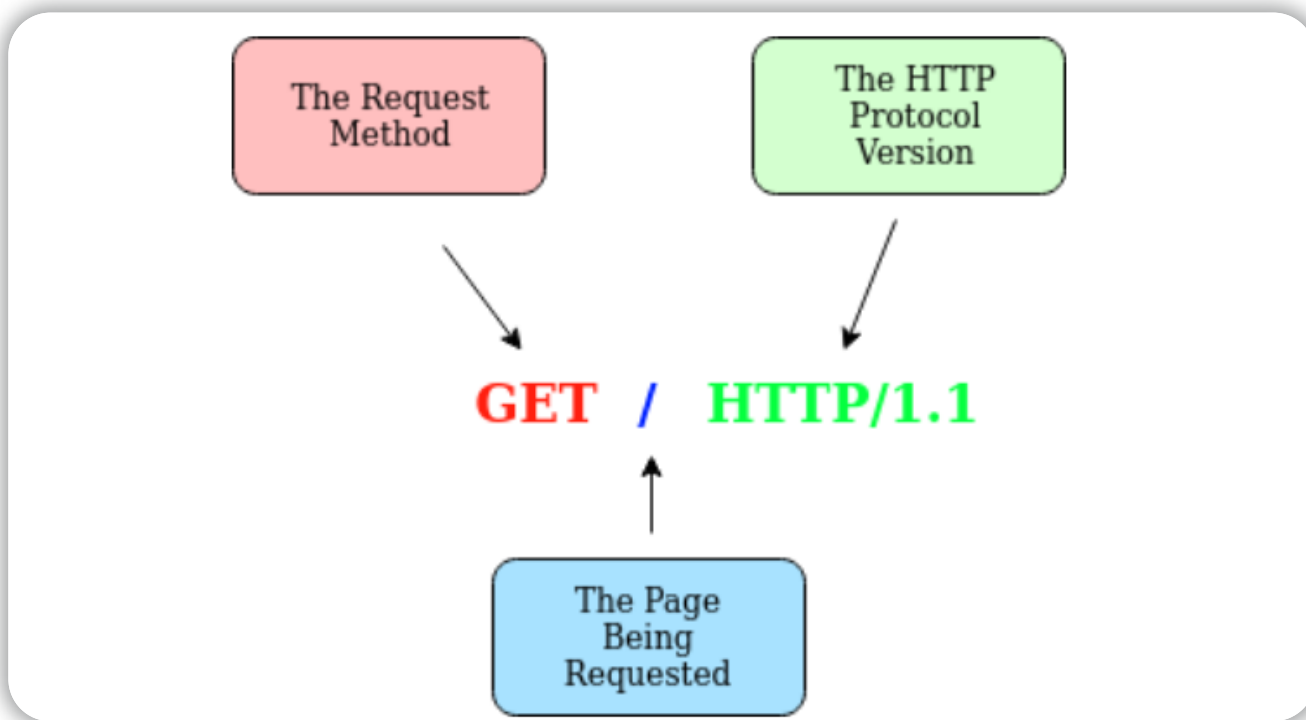


- **Scheme**: 这说明了使用什么协议来访问资源，如HTTP、HTTPS、FTP(文件传输协议)。
- **User**: 有些服务需要认证以完成登录，你可以在URL中输入用户名和密码进行登录。
- **Host/Domain**: 你希望访问的服务器的域名或IP地址。
- **Port**: 你将要连接到的端口，通常HTTP协议将使用 80 端口，HTTPS协议将使用 443 端口，但是这些协议也可以选择使用 1 - 65535 之间的任何端口号。
- **Path**: 你试图访问的资源的文件名或位置。
- **Query String**: 可以发送到请求路径的额外信息位。例如，`/blog?id=1` 将告诉blog path你希望接收id为1的博客文章。
- **Fragment**: 这是对所请求的实际页面上的位置的引用。这通常用于具有较长内容的页面，并且可以将页面的某一部分直接链接到该引用；因此，只要用户通过该URL访问页面，就可以看到链接所对应的

部分内容。

发出请求

只需要一行"GET / HTTP/1.1"就可以向web服务器发出请求。



但为了获得更丰富的网络体验，你还需要通过请求消息发送其他数据，这些数据将通过请求报头进行发送，在请求报头中也会包含一些额外的信息，以提供给你进行通信的web服务器。

请求示例（由浏览器客户端发送至服务器端）

```
GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Referer: https://tryhackme.com/
```

分解以上请求示例的每一行:

- 第1行: 这个请求将发送GET方法(详见本文的HTTP方法小节), 用 / 请求主页, 并告诉web服务器我们正在使用的HTTP协议版本为 1.1 。
- 第2行: 我们将告诉web服务器我们想要访问域名为 tryhackme.com 的网站。
- 第3行: 我们将告诉web服务器我们正在使用Firefox 87版本浏览器。

- 第4行：我们正在告诉web服务器，将我们指向当前页面的来源网址（引用页）是
`https://tryhackme.com`
- 第5行：HTTP请求总是以空行结束，以通知web服务器该HTTP请求已经完成。

响应示例（由服务器端发送至浏览器客户端）

```
HTTP/1.1 200 OK
Server: nginx/1.15.8
Date: Fri, 09 Apr 2021 13:34:03 GMT
Content-Type: text/html
Content-Length: 98

<html>
<head>
  <title>TryHackMe</title>
</head>
<body>
  Welcome To TryHackMe.com
</body>
</html>
```

分解以上响应示例的每一行:

- 第1行：HTTP 1.1是服务器正在使用的HTTP协议版本，然后是HTTP状态码(在本例中为“200 Ok”)，它告诉我们该响应对应的请求已经成功完成。
- 第2行：这告诉我们web服务器使用的软件和其版本号。
- 第3行：web服务器的当前日期、时间和时区。
- 第4行：Content-Type报头会告诉客户端该web服务器将要发送什么类型的信息，比如HTML、图像、视频、pdf、XML。
- 第5行：Content-Length将告诉客户端该响应的长度，这样我们就可以确认没有发生数据丢失。
- 第6行：HTTP响应包含一个空行，用于确认HTTP响应的结束。
- 第7-14行：被请求的信息，在本例中是关于目标主页的html代码。

答题

回答以下问题

上例中使用的是什么 HTTP 协议？

HTTP/1.1

正确答案

什么响应标头告诉浏览器需要多少数据？

Content-Length

正确答案

H2 HTTP方法

HTTP 方法是客户端在发出 HTTP 请求时显示其预期操作的一种方式。有很多 HTTP 方法，在此我们将介绍最常见的方法，在大多数情况下你将处理的是 GET 和 POST 方法。

GET 请求：用于从 Web 服务器中获取信息。

POST 请求：用于向 Web 服务器提交数据并可能创建新记录。

PUT 请求：用于向 Web 服务器提交数据以更新信息。

DELETE 请求：用于从Web服务器中删除信息/记录。

答题

回答以下问题

将使用什么方法来创建新的用户帐户？

POST

正确答案

将使用什么方法更新您的电子邮件地址？

PUT

正确答案

将使用什么方法删除您上传到帐户的图片？

DELETE

正确答案

将使用什么方法来查看新闻文章？

GET

正确答案

H2 HTTP状态代码

HTTP状态码

在前面的叙述中，你能了解到当 HTTP 服务器发出响应时，第一行总是会包含一个状态代码，该状态码用于通知客户端所发出请求对应的结果以及针对请求的可能处理方式。 这些状态码可以分为 5 个不同的范围：

100-199 - 信息响应	发送这些消息是为了告诉客户他们的请求的第一部分已被接受，他们应该继续发送请求的其余部分。这些代码不常见。
200-299 - 成功	此范围的状态代码用于告诉客户端他们的请求已成功。
300-399 - 重定向	用于将客户端的请求重定向到另一个资源。可以是不同的网页或完全不同的网站。
400-499 - 客户端错误	用于通知客户他们的请求有错误。
500-599 - 服务器错误	这是为服务器端发生的错误保留的，通常表示服务器处理请求时出现了相当大的问题。

常见的HTTP状态码

有很多不同的 HTTP 状态码，这还不包括应用程序自己定义的状态码，我们将介绍一些你可能遇到的最常见的 HTTP 响应状态码：

200 - 确定	请求已成功完成。
201 - 创建	已创建资源（例如新用户或新博客文章）。
301 - 永久重定向	这会将客户端的浏览器重定向到一个新网页或告诉搜索引擎该页面已移动到其他地方。
302 - 临时重定向	类似于上面的永久重定向，但顾名思义，这只是暂时的改变，不久的将来可能会再次改变。
400 - 错误请求	这告诉浏览器他们的请求中有错误或缺失。如果正在请求的 Web 服务器资源需要客户端未发送的特定参数，则有时可以使用此状态码。
401 - 未授权	目前不允许您查看此资源，直到您通过网络应用程序获得授权，最常见的是使用用户名和密码。
403 - 禁止	无论您是否登录，您都无权查看此资源。
405 - 方法不允许	资源不允许此方法请求，例如，当它期望 POST 请求时，您向资源 /create-account 发送了 GET 请求。
404 页面不存在	您请求的页面/资源不存在。
500 - 内部服务错误	服务器在您的请求中遇到了某种错误，它不知道如何正确处理。
503 服务不可用	此服务器无法处理您的请求，因为它超载或停机维护。

答题

回答以下问题

如果您创建了新用户或博客文章，您会收到什么响应代码？

201

正确答案

如果您尝试访问不存在的页面，您可能会收到什么响应代码？

404

正确答案

如果 Web 服务器无法访问其数据库并且应用程序崩溃，您可能会收到什么响应代码？

503

正确答案

如果您尝试在不先登录的情况下编辑您的个人资料，您可能会收到什么响应代码？

401

正确答案

H2 请求标头和响应标头

标头是你在发出请求时可以发送到 Web 服务器的额外数据位。

虽然在发出 HTTP 请求时没有严格要求标头，但在缺少标头的情况下，你会发现你很难正确地查看网站。

常见的请求头

以下是从客户端（通常是你的浏览器）发送到服务器端的标头信息。

Host: 一些 Web 服务器可能会在多个网站上托管内容，因此通过提供主机标头，你可以告诉web服务器你需要访问哪个网站，否则你只会收到web服务器对应的默认网站的响应。

User-Agent: 这是你所使用的浏览器软件类型和版本号，告诉web服务器你的浏览器软件类型能帮助它为你的浏览器正确格式化网站，而且网站相关的 HTML、JavaScript 和 CSS 的一些元素只在某些浏览器中可用。

Content-Length: 当向 Web 服务器发送数据时，例如当浏览器通过表单向web服务器发送数据时，Content-Length将告诉 Web 服务器该 Web 请求所期望的数据长度，这样web服务器就可以确保它在响应浏览器请求时并不会丢失任何数据。

Accept-Encoding: 这将告诉web服务器 当前使用的浏览器支持什么类型的压缩方法，这样数据就可以变小以便通过互联网进行传输。

Cookie: Cookie 是发送到服务器以帮助记住你的信息的数据。

常见的响应头

以下是发出请求后从服务器端返回给浏览器客户端的标头信息。

Set-Cookie: 表示浏览器端要存储的Cookie信息，之后每次浏览器发出请求时，Cookie值都会发送回 Web 服务器。

Cache-Control: 在再次请求之前，响应内容在浏览器缓存中将存储多长时间。

Content-Type: 这将告诉客户端 从web服务器端所返回的数据类型，即 HTML、CSS、JavaScript、图像、PDF、视频等。使用Content-Type标头，浏览器才能知道如何处理数据。

Content-Encoding: 在通过 Internet 发送数据时，web服务器将使用什么方法压缩数据以使其更小。

答题

回答以下问题

什么标头告诉网络服务器正在使用什么浏览器？

User-Agent

正确答案

什么标头告诉浏览器正在返回什么类型的数据？

Content-Type

正确答案

什么标头告诉 Web 服务器正在请求哪个网站？

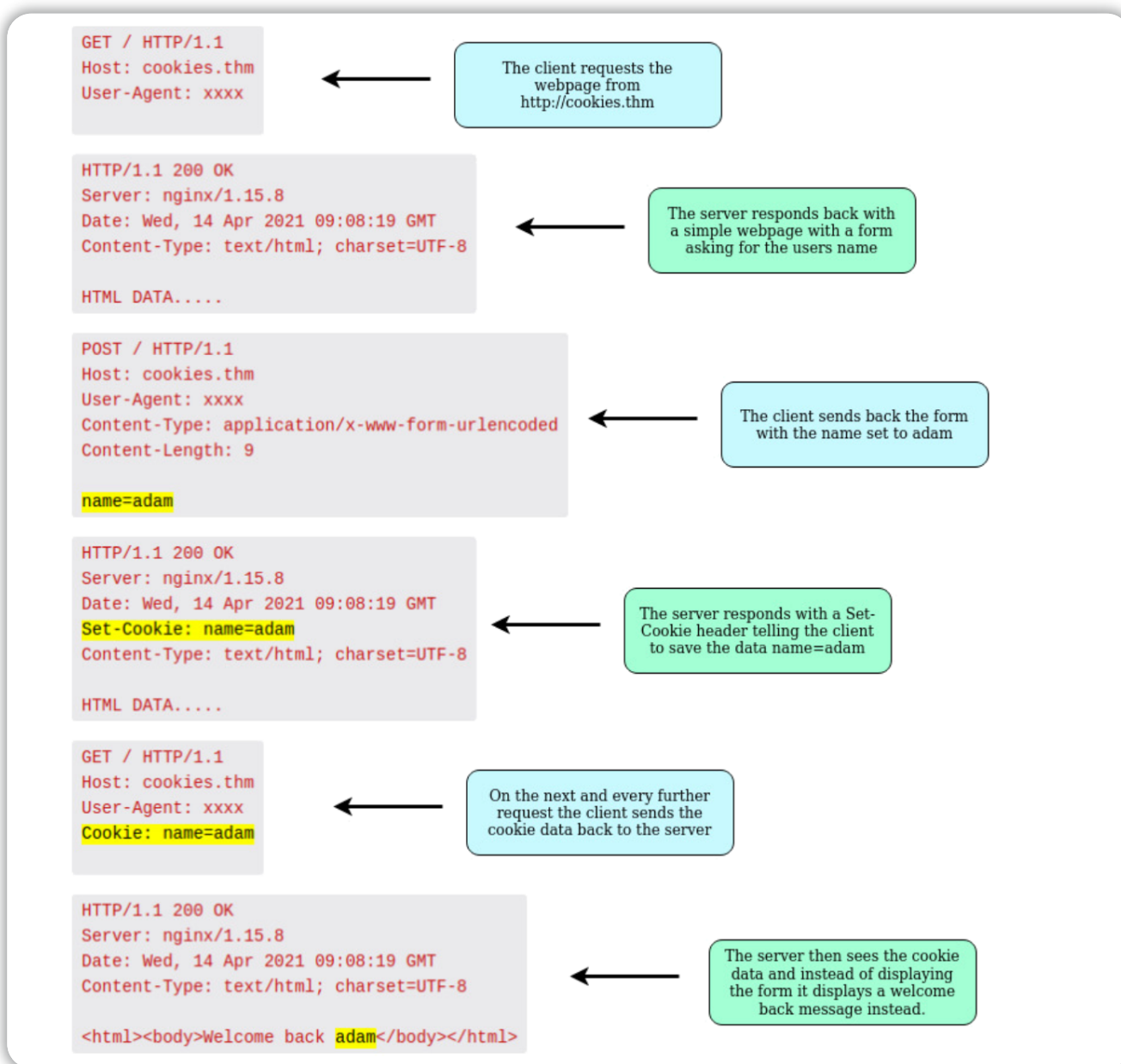
Host

正确答案

H2 Cookies

Cookie是存储在你的计算机上的一小段数据。当你从web服务器收到“Set-Cookie”标头时，对应的Cookie信息将被浏览器保存，然后，你发出的每一个进一步请求，都会将Cookie数据发送回web服务器。由于HTTP是无状态的（不跟踪你之前的请求），所以Cookie可用于提醒web服务器你的身份、网站的一些个人设置或者你以前是否访问过该网站。

让我们看一下下面这个HTTP请求示例：



Cookie 可用于多种用途，但最常用于网站身份验证。cookie 值通常不会是一个可以让你直接看到密码的明文字符串，而会是一个令牌-token（具有不容易被猜到的唯一密码）。

查看你的 Cookie

你可以使用浏览器中的开发者工具查看你的浏览器向网站发送了哪些 cookie。

打开开发人员工具后，单击“网络”选项卡。此选项卡将向你显示你的浏览器已请求的所有资源的列表，你可以单击每一个子项以查看请求和响应的详细分类信息。如果你的浏览器发送了 cookie，你将在请求消息的“Cookie”选项卡上看到具体的cookie内容。

答题

回答以下问题

哪个标头用于将 cookie 保存到您的计算机？

Set-Cookie

正确答案

H2

发出请求

答题

回答以下问题

向 /room 发出 GET 请求

THM{YOU'RE_IN_THE_ROOM}

正确答案

暗示

向 /blog 发出 GET 请求，并使用齿轮图标在 URL 字段中将 id 参数设置为 1

THM{YOU_FOUND_THE_BLOG}

正确答案

向 /user/1 发出 DELETE 请求

THM{USER_IS_DELETED}

正确答案

将用户名参数设置为 admin 向 /user/2 发出 PUT 请求

THM{USER_HAS_UPDATED}

正确答案

暗示

将 thm 的用户名和 letmein 的密码发布到 /login

THM{HTTP_REQUEST_MASTER}

正确答案

暗示

GET

http://tryhackme.com/

⚙️

Go

GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0

Change your HTTP request method

Change your URL here

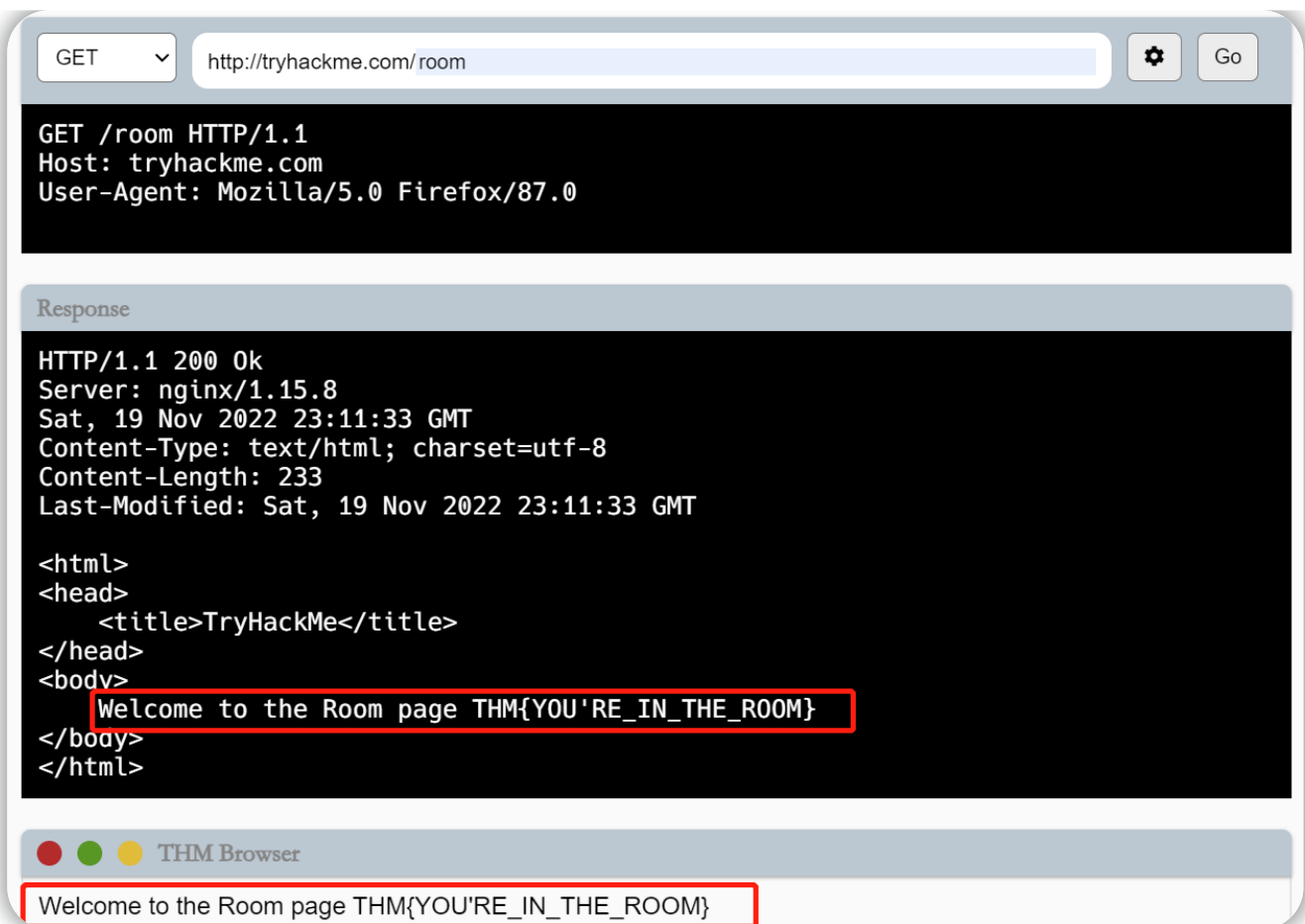
Update your GET & POST parameters here

Click here to send your request

Response

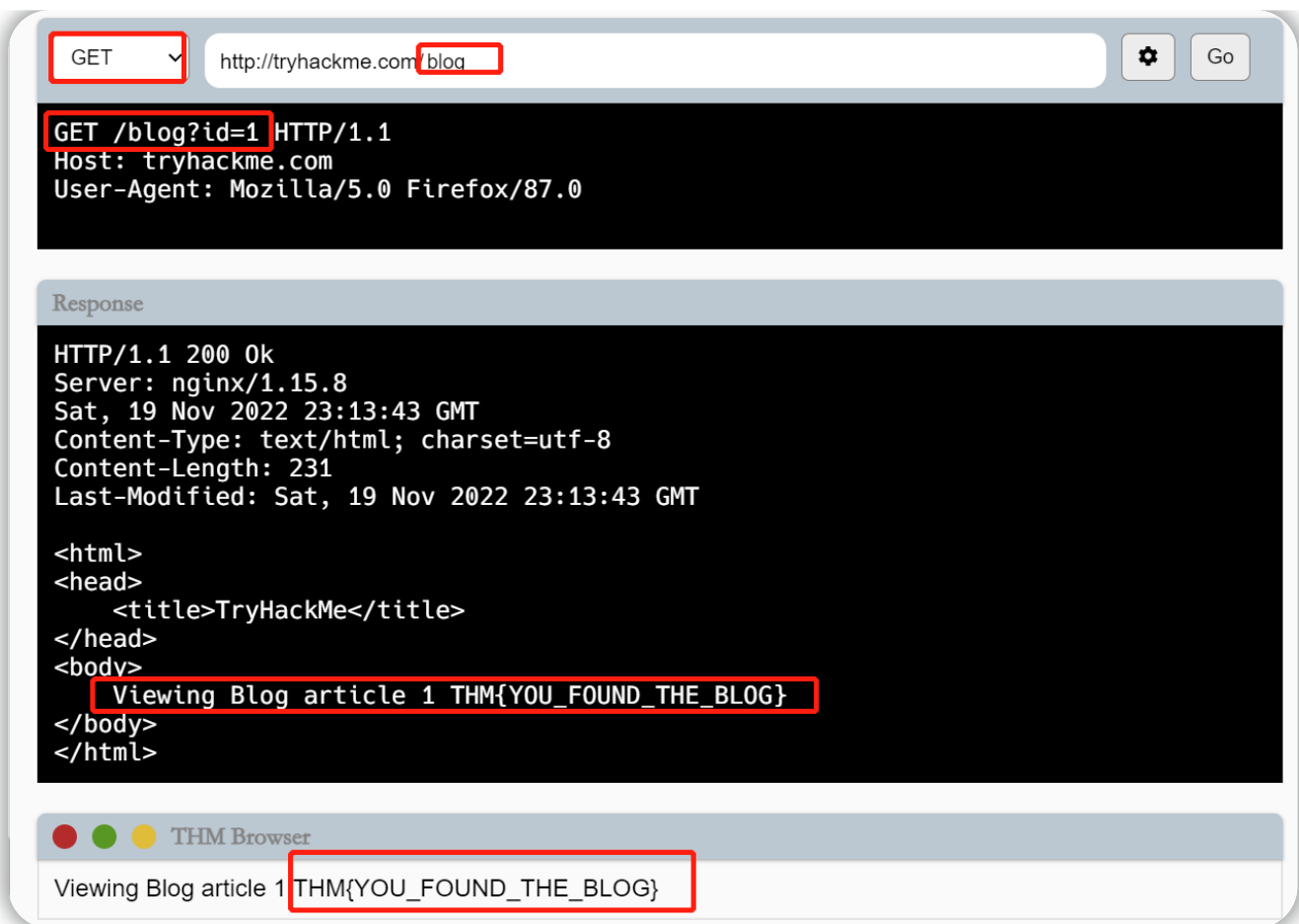
THM Browser

问题一：

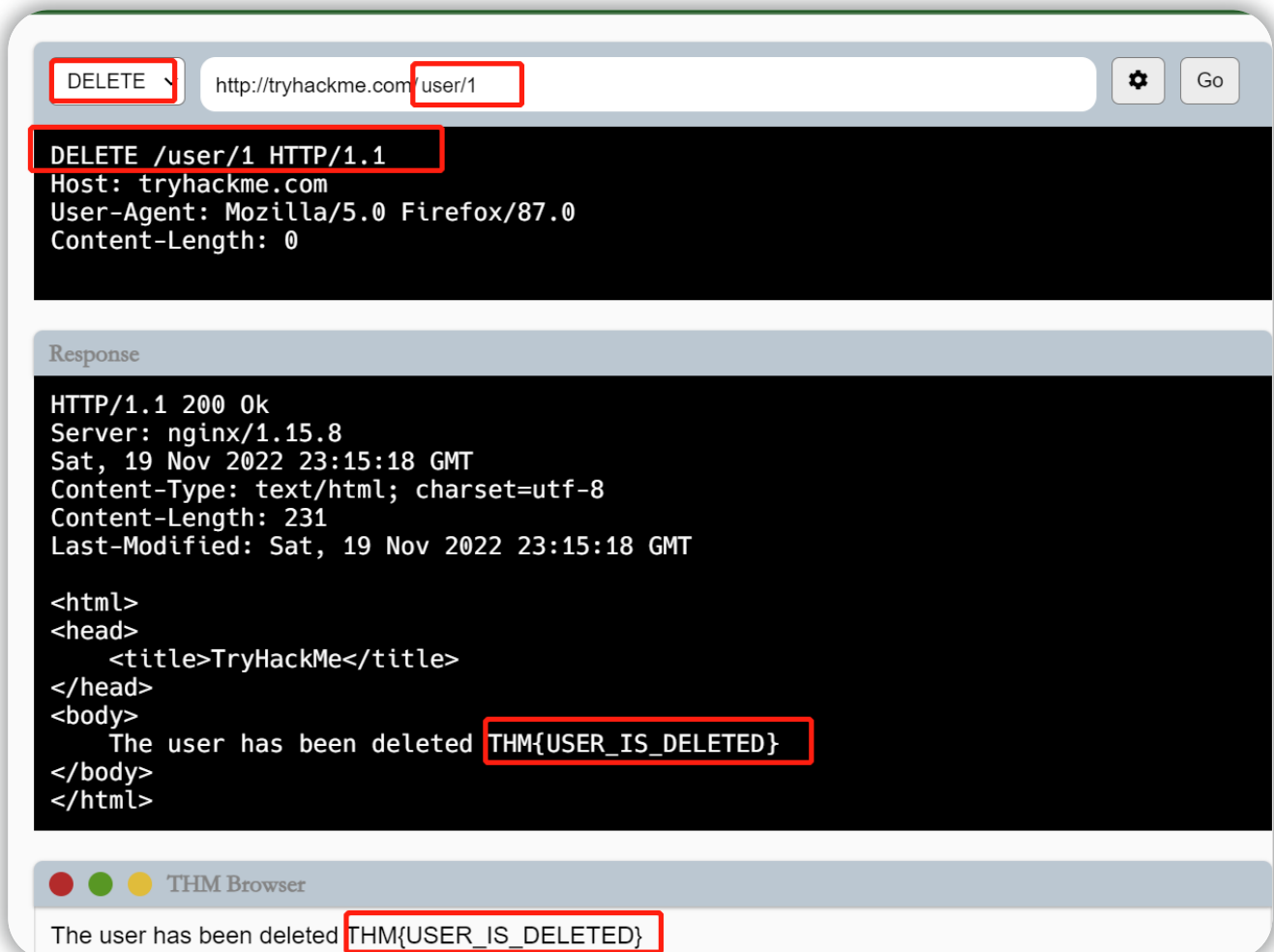


问题二：







问题三：





问题四：

PUT  Go

PUT Parameters 

PUT /user/2 HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0
Content-Length: 0

username = admin 

PUT  Go

PUT /user/2 HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 14

username=admin

Response


HTTP/1.1 200 Ok
Server: nginx/1.15.8
Sat, 19 Nov 2022 23:19:17 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 232
Last-Modified: Sat, 19 Nov 2022 23:19:17 GMT

<html>
<head>
 <title>TryHackMe</title>
</head>
<body>
 Username changed to admin THM{USER_HAS_UPDATED}
</body>
</html>

THM Browser

Username changed to admin THM{USER_HAS_UPDATED}

问题五：

POST ☒  Go

