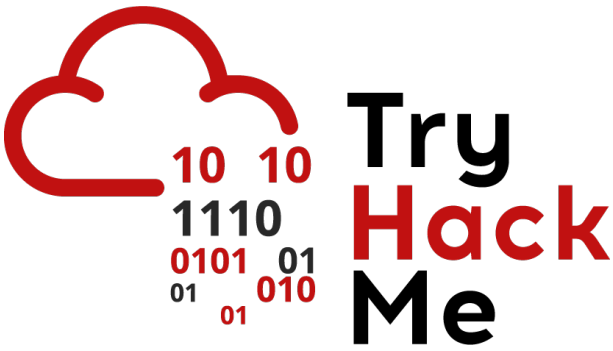
Writeup: Sala *HTTP in Detail*

Autor: Ismaeldevs

Plataforma: TryHackMe

4 de julio de 2025



Índice

1. Introducción

En esta sala aprenderemos cómo funciona **HTTP**, cómo se estructuran las solicitudes y respuestas, qué son los métodos HTTP y cómo analizarlos. También se exploran los códigos de estado, encabezados importantes y cómo HTTPS mejora la seguridad mediante cifrado.

2. Sala

2.1. Tarea 1 - Qué es HTTP(S)?

Nos adentramos en los conceptos básicos del protocolo **HTTP (HyperText Transfer Protocol)**, el cual es utilizado para la transmisión de datos en la web. Además, explica cómo los navegadores y servidores se comunican usando este protocolo, y qué significa cuando una URL comienza con **http:**// o **https:**//, entre otras diferencias.

Una vez que aprendemos los conceptos de HTTP podemos resolver la tarea respondiendo las siguientes preguntas:

Pregunta: What does HTTP stand for?
Respuesta: **HyperText Transfer Protocol**

Pregunta: What does the S in HTTPS stand for?
Respuesta: **Secure**

Pregunta: On the mock webpage on the right there is an issue, once you've found it, click on it. What is the challenge flag?

Para resolver este último debemos desplegar el sitio haciendo clic en **View Site**.

Una vez desplegado analizaremos el sitio y notaremos que no es seguro, por ende, el error es que la página web no es segura.

Haremos clic en el candado y nos saltara una alerta con la flag de respuesta que estamos buscando para completar la tarea.

Respuesta: **THM{INVALID_HTTP_CERT}**

2.2. Tarea 2 - Solicitudes y respuestas

En esta tarea vamos a profundizar en la estructura de las solicitudes (**requests**) y respuestas (**responses**) HTTP. Aprenderemos cómo un navegador envía una solicitud al servidor especificando un método (**GET** o **POST**), una ruta y una versión del protocolo, acompañados de encabezados (**headers**).

Al entender la estructura de las solicitudes y respuestas podemos responder las siguientes preguntas para completar la tarea.

Pregunta: What HTTP protocol is being used in the above example?
Respuesta: **HTTP/1.1**

Pregunta: What response header tells the browser how much data to expect?
Respuesta: **Content-Length**

2.3. Tarea 3 - Métodos HTTP

Aquí nos vamos a enfocar en los diferentes métodos HTTP donde cada método tiene un propósito específico, y el comprenderlos es esencial para interactuar correctamente con aplicaciones web, algunos métodos son:

- **GET**
- **POST**
- **PUT**
- **DELETE**

Después de comprender la tarea de los métodos HTTP, vamos a responder las siguientes preguntas.

Pregunta: What method would be used to create a new user account?
Respusta: **POST**

Pregunta: What method would be used to update your email address?
Respusta: **PUT**

Pregunta: What method would be used to remove a picture you've uploaded to your account?
Respusta: **DELETE**

Pregunta: What method would be used to view a news article?
Respusta: **GET**

2.4. Tara 4 - Códigos de estado HTTP

En esta tarea abordaremos los códigos de estado HTTP, estos códigos de estado son respuestas númericas por el servidor para indicar el resultado de una solicitud.

Ahora, pasaremos a responder las siguientes preguntas.

Pregunta: What response code might you receive if you've created a new user or blog post article?
Respuesta: **201**

Pregunta: What response code might you receive if you've tried to access a page that doesn't exist?
Respuesta: **404**

Pregunta: What response code might you receive if the web server cannot access its database and the application crashes?
Respuesta: **503**

Pregunta: What response code might you receive if you try to edit your profile without logging in first?
Respuesta: **401**

2.5. Tarea 5 - Encabezados

Aprenderemos sobre los encabezados (**headers**) HTTP, que son campos **clave-valor** incluidos en solicitudes y respuestas para transmitir información adicional sobre la comunicación. Estos encabezados permiten controlar aspectos como la autenticación, el tipo de contenido, el control de caché y más.

Algunos encabezados importantes que se abordan son:

- **User-Agent**
- **Host**
- **Content-Type**
- **Authorization**

Una vez que comprendemos como funcionan los encabezados podemos responder las siguientes preguntas.

Pregunta: What header tells the web server what browser is being used?
Respuesta: **User-Agent**

Pregunta: What header tells the browser what type of data is being returned?
Respuesta: **Content-Type**

Pregunta: What header tells the web server which website is being requested?
Respuesta: **Host**

2.6. Tarea 6 - Cookies

Vamos a conocer qué son las **cookies HTTP**, las cuales se dicen que son pequeños fragmentos de datos que el servidor envía al navegador para que los almacene y reenvíe en futuras solicitudes. Las cookies permiten mantener sesiones, guardar preferencias del usuario y realizar seguimiento de actividad.

Después de aprender sobre las Cookies podemos responder la siguiente pregunta.

Pregunta: Which header is used to save cookies to your computer?
Respuesta: **Set-Cookie**

2.7. Tarea 7 - Realizar Solicitudes

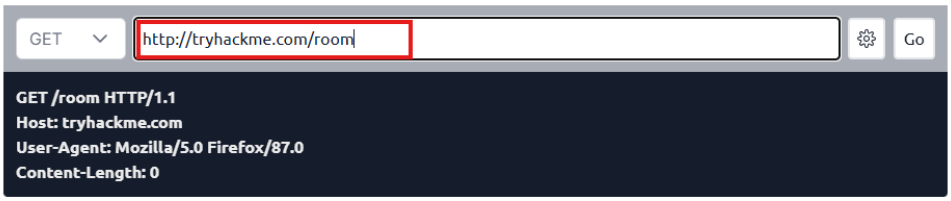
Para completar nuestro aprendizaje en esta sala, debemos realizar un ejercicio donde desplegaremos un emulador para hacer las solicitudes HTTP.

Para comenzar vamos a hacer clic en **View Site** que se encuentra en el lado superior.

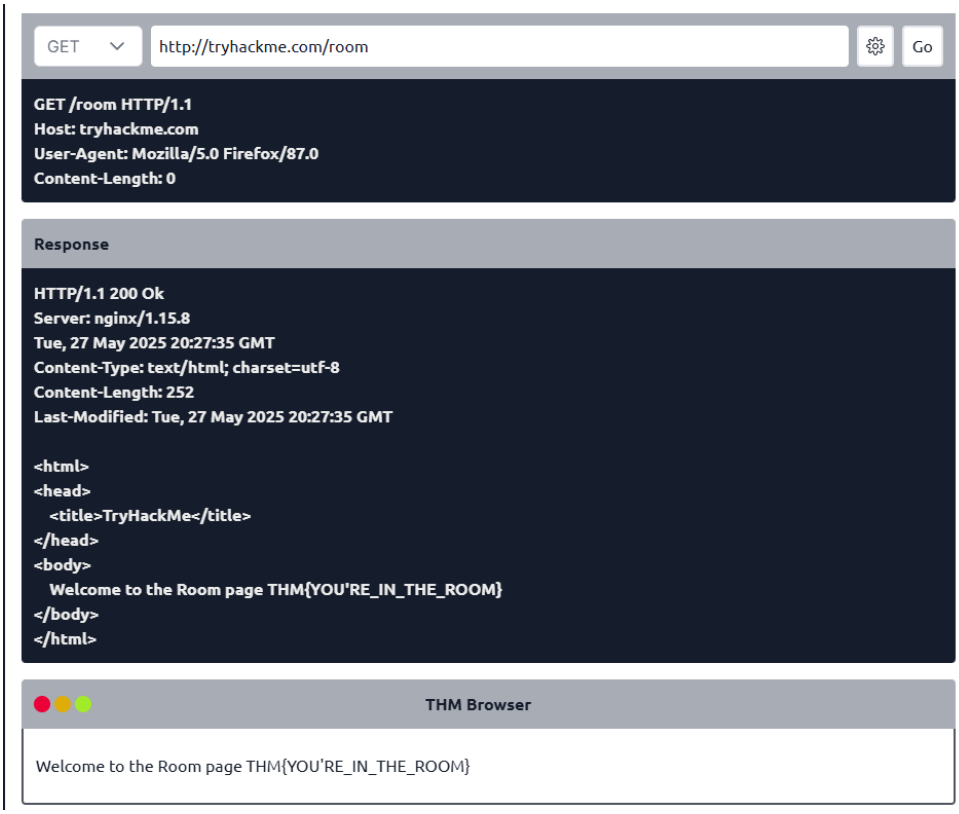
Se nos desplegará la página donde empezaremos a realizar las solicitudes HTTP para lograr obtener las flags de respuesta de lo siguiente:

Make a GET request to /room page

Para completar este primer reto debemos ir a la URL del sitio y agregar en la URL la ruta /**room**.



Una vez que ingresamos la ruta /**room** vamos a darle al botón de **Go** y nos lanzará de resultado la flag de respuesta de la primera solicitud.

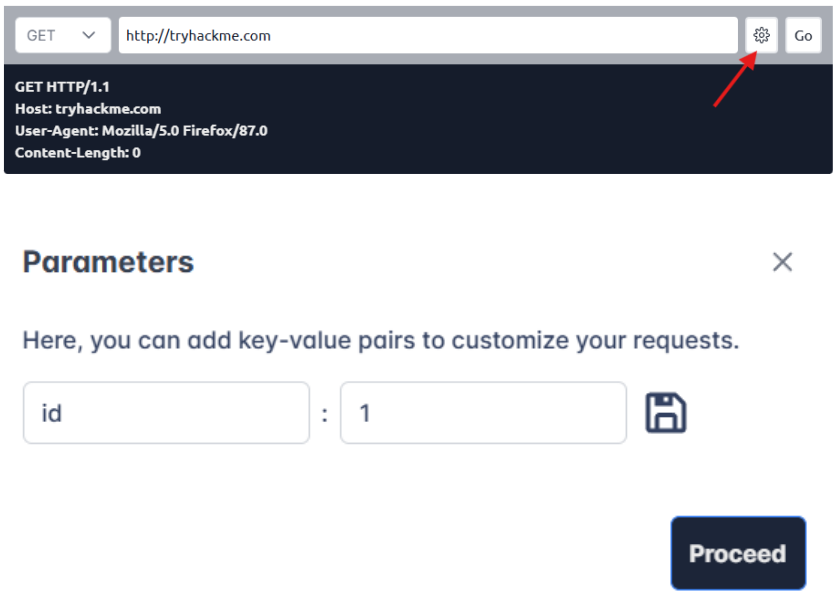


Respuesta: THM{YOU'RE_IN_THE_ROOM}

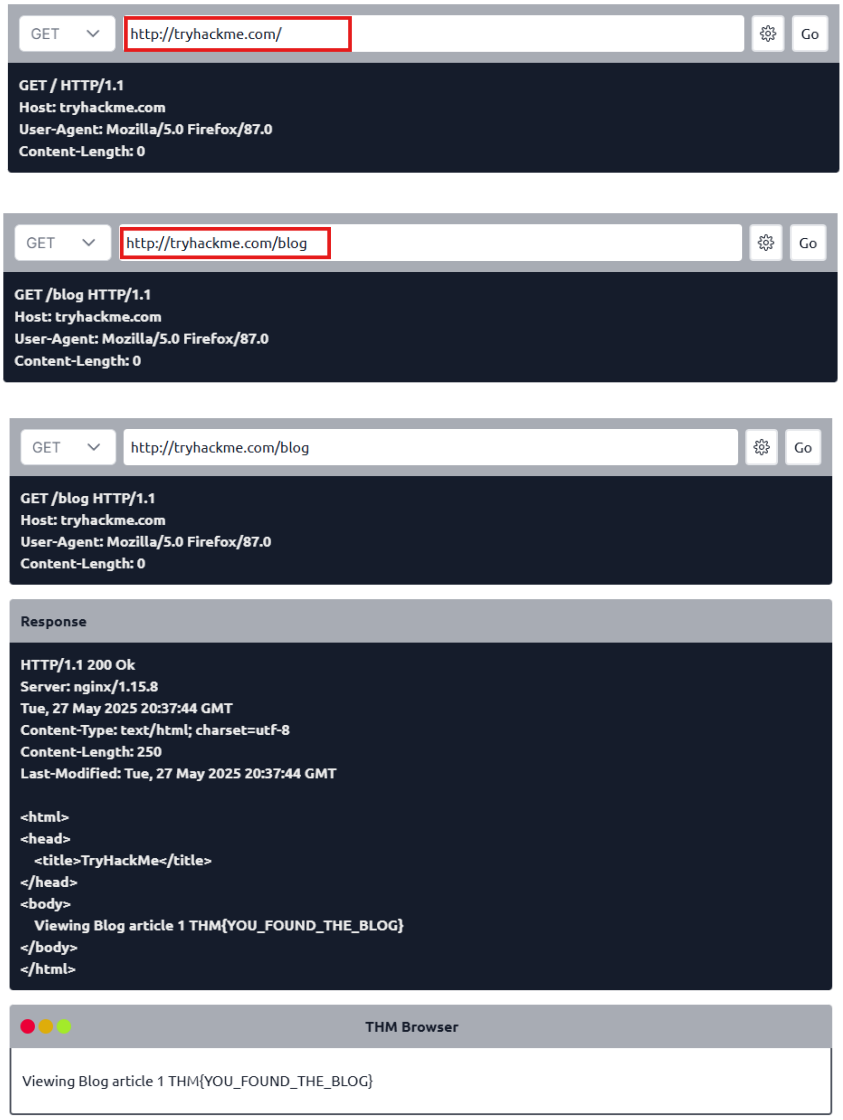
La siguiente solicitud que debemos realizar es:

Make a GET request to /blog page and set the id parameter to 1 Note: Use the gear button on the right to manage URI parameters

Para completarlo debemos ir primero al icono de engranaje y colocar los parametros requeridos para poder hacer la solicitud.



Una vez colocado los parámetros y haberlos guardado, tendremos que ir a la URL y colocar la ruta de /**blog**, posterior a eso daremos clic en **Go** y saltará los resultados con la flag que necesitamos.



Respuesta: **THM{YOU_FOUND_THE_BLOG}** Ahora, la siguiente solicitud nos pide:

Make a DELETE request to /user/1 page

Para completar esta sencilla solicitud HTTP debemos dirigirnos a la URL y colocar la ruta /**user/1** sin la necesidad de colocar parámetros, después procederemos a cambiar el tipo de solicitud por **DELETE** y haremos clic en **Go** para realizar la petición y nos devuelva como respuesta la flag.

Ahora, la siguiente solicitud por completar es

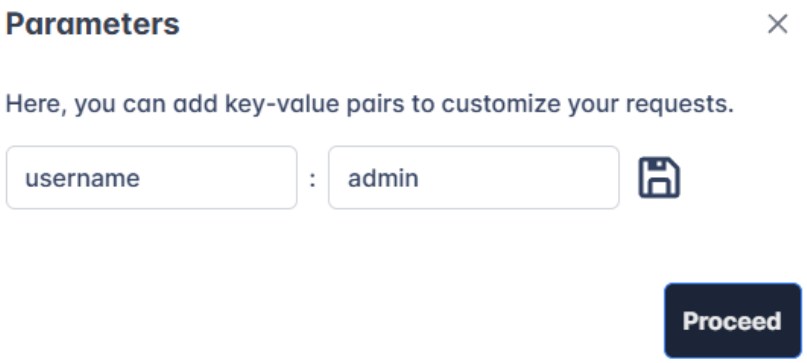
Make a PUT request to /user/2 page with the username parameter set to admin Note: Use the gear button on the right to manage body parameters

En este caso debemos colocar parámetros, cambiar el tipo de solicitud HTTP y añadir la ruta en la URL. Para ello, lo primero que haremos es añadir los parámetros correspondientes:

- **username** | **admin**

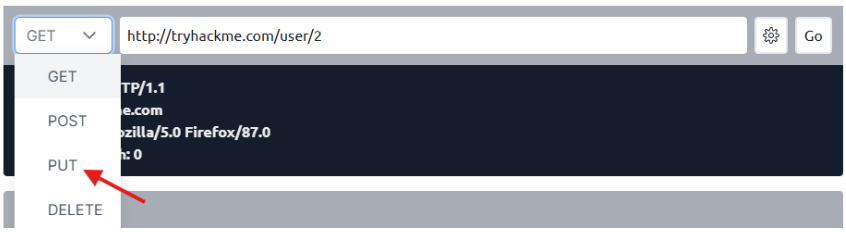
Una vez que guardamos los parámetros, vamos a añadir la ruta /**user/2** en la URL, después cambiamos el tipo de solicitud a **PUT** y para finalizar le daremos al botón de **Go** para enviar la solicitud y nos responda con la flag.

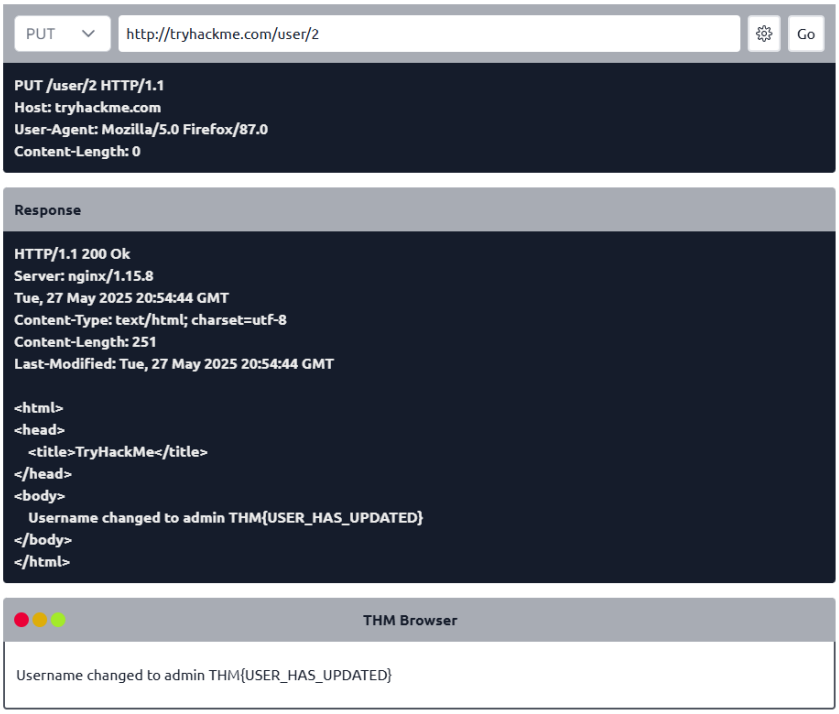

GET
http://tryhackme.com
Go
GET HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0


Parameters
Here, you can add key-value pairs to customize your requests.
username
:
admin
Proceed


GET
http://tryhackme.com/
Go
GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0


GET
http://tryhackme.com/user/2
Go
GET /user/2 HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0


GET
http://tryhackme.com/user/2
Go
GET
POST
PUT
DELETE



Respuesta: THM{USER_HAS_UPDATED}

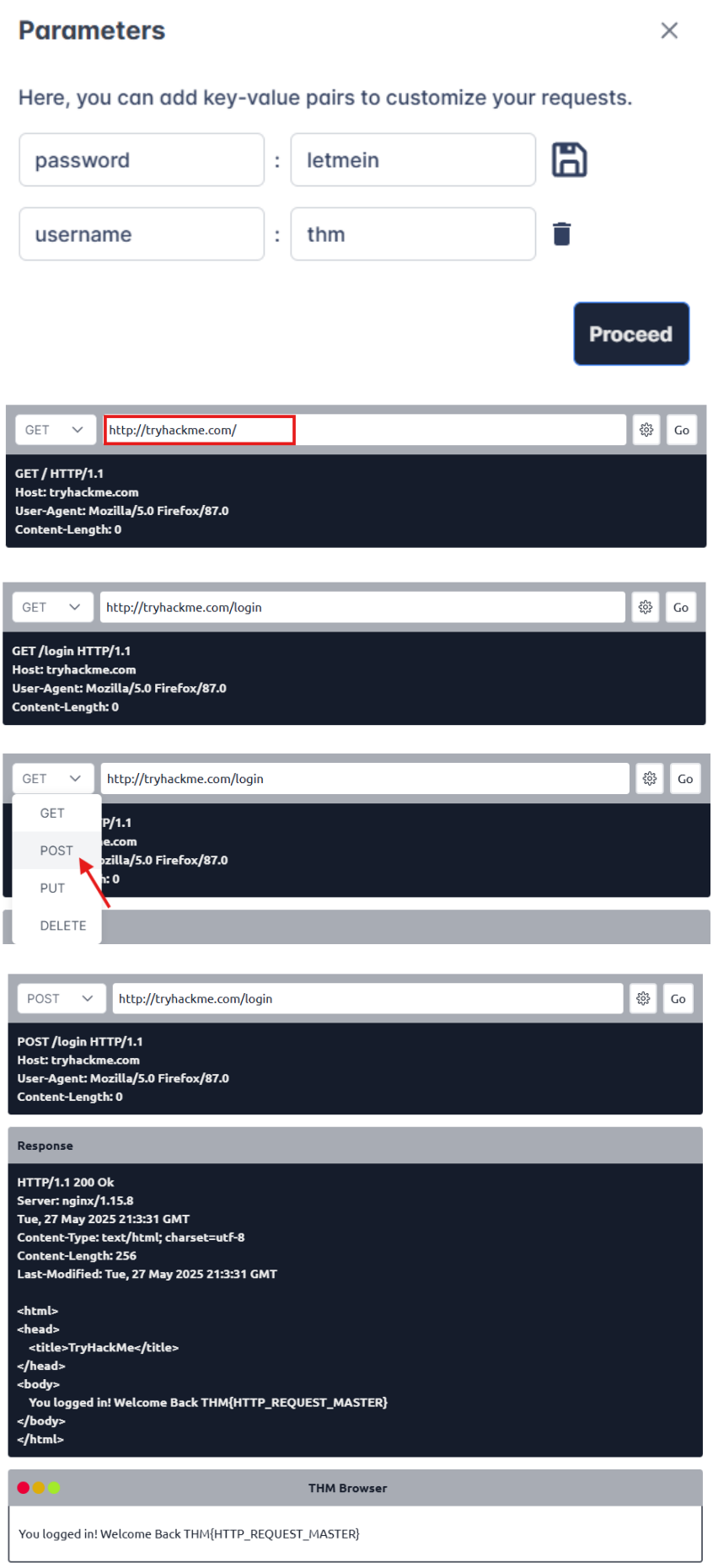
Ahora, realizaremos la última petición de esta tarea para completar la sala:

Make a POST request to /login page with the username of thm and a password of letmein Note: Use the gear button on the right to manage body parameters

Para completar este último debemos hacer los mismos pasos de la anterior petición pero con la diferencia que tenemos que añadir dos parámetros más:

- **username** | **thm**
- **password** | **letmein**

Después de añadir los parámetros, vamos a implementar la ruta /**login** en la URL y cambiamos el tipo de petición por **POST** y terminamos haciendo clic en **Go** para enviar la solicitud y nos responda con la flag.


Parameters
Here, you can add key-value pairs to customize your requests.
password : letmein
username : thm
Proceed
GET http://tryhackme.com/ Go
GET / HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0
GET http://tryhackme.com/login Go
GET /login HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0
GET http://tryhackme.com/login Go
GET
POST
PUT
DELETE
POST http://tryhackme.com/login Go
POST /login HTTP/1.1
Host: tryhackme.com
User-Agent: Mozilla/5.0 Firefox/87.0
Content-Length: 0
Response
HTTP/1.1 200 Ok
Server: nginx/1.15.8
Tue, 27 May 2025 21:3:31 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 256
Last-Modified: Tue, 27 May 2025 21:3:31 GMT
<html>
<head>
<title>TryHackMe</title>
</head>
<body>
You logged in! Welcome Back THM{HTTP_REQUEST_MASTER}
</body>
</html>
THM Browser
You logged in! Welcome Back THM{HTTP_REQUEST_MASTER}

Respuesta: THM{HTTP_REQUEST_MASTER}

3. Conclusión sobre la Sala

Esta sala nos ayudo a comprender el funcionamiento del protocolo HTTP, el cual es un elemento esencial en la comunicación entre clientes y servidores web. A lo largo de la sala se aprendió a identificar y analizar solicitudes y respuestas HTTP, los métodos más utilizados, los códigos de estado, encabezados y el manejo de cookies.