The Internet Is Physical

- **Not a "Cloud"**: The internet is a network of physical wires (fiber optics, copper, etc.) buried underground or transmitted via satellites/cell towers.
- **Servers vs. Clients**:
  - **Servers**: Directly connected to the internet, store data (e.g., websites), and have unique IP addresses (e.g., `72.14.205.100`).
  - **Clients**: Devices (e.g., laptops, phones) connect *indirectly* via ISPs (e.g., DSL, dial-up).

Communication Basics

- **Data as Packets**: Information (emails, web pages) is split into smaller **packets**, sent across the internet, and reassembled at the destination.
- **IP Addresses**: Unique identifiers for every device/server (like postal addresses). Names like `google.com` simplify complex IPs for humans.

How Data Travels

- **Routers**: Direct packets between networks. Each router adds a "layer" (its IP) to packets, guiding them step-by-step to their destination.
  - Example: Visiting `aol.com` involves 10–15 routers to route packets.
- **Privacy**: Routers use IP addresses to ensure packets reach the correct device (e.g., your laptop, not your boss's).

Real-World Example

- **Sending an Email**:
  1. Compose in Gmail (client).
  2. Gmail (server) breaks the email into packets.
  3. Routers guide packets to Aunt Ruth's AOL server.
  4. Ruth retrieves the email via her ISP (dial-up).

How the Web Works

- **Requesting a Web Page**
  - When a user enters a URL in the browser, the browser contacts a **DNS (Domain Name System) server** to find the corresponding **IP address** of the website.
  - A request is then sent to the web server that hosts the site.
- **Server Response & Rendering**
  - The web server processes the request and sends back a response, which can be a **static** page (pre-written content) or a **dynamic** page (generated on demand).
  - The response includes:
    - **HTML** – Defines the page structure.
    - **CSS** – Styles the page for better design.
    - **JavaScript** – Adds interactivity and dynamic features.
- **Protocols & Security**
  - **HTTP (Hypertext Transfer Protocol)** governs communication between the browser and the server.
  - **HTTPS** encrypts the data to ensure security and privacy, preventing unauthorized access.
- **Web Technologies**
  - **Frontend**: Uses HTML, CSS, and JavaScript to display and interact with content.
  - **Backend**: Uses languages like Node.js, PHP, and Python to process data and handle requests.
  - **APIs (Application Programming Interfaces)** allow websites and applications to exchange data, often using **JSON (JavaScript Object Notation)** format.

- **Advanced Features**
    - **WebSockets** enable real-time communication for applications like live chat and stock market updates.
    - **Frameworks** simplify development, improving efficiency and performance for both frontend and backend.

Client-Server Model & Peer-to-Peer Model

- **What is the Client-Server Model?**
- The **client-server model** is a web architecture that divides computers into two roles:
    - **Clients**: Devices or programs that request services (e.g., web browsers, mobile apps).
    - **Servers**: Programs that provide requested services (e.g., web servers, database servers).
- Communication happens through the **request-response cycle**, usually via **HTTP**.
- **How Servers Work**
- A **server** is a program (not just a powerful computer) that runs continuously to handle client requests.
- One server can handle multiple clients simultaneously.
- Types of servers:
    - **Web servers** (e.g., Apache, Nginx) serve websites.
    - **Database servers** (e.g., MySQL, PostgreSQL) store and manage data.
- **3. Peer-to-Peer (P2P) Model – The Alternative**
- In the **peer-to-peer (P2P) model**, there is no fixed client or server—every computer can send and receive data.

- This model is **decentralized**, unlike the client-server model, which is centralized.
- Examples: **BitTorrent, video chat applications**.

. Key Differences

Feature	Client-Server Model	Peer-to-Peer Model
Structure	Centralized	Decentralized
Role of Devices	Clients request, servers provide	All devices act as both
Example	Websites, cloud services	File sharing, VoIP apps

- **HTTP Basics**
  - **Definition**: HTTP is a stateless protocol for client-server communication. Each request is independent (no memory of prior interactions).
  - **HTTPS**: Secure version of HTTP using SSL/TLS encryption for sensitive data (e.g., passwords, payments).

- **HTTP Methods**
  - **GET**: Fetch data (e.g., load a webpage, retrieve JSON).
  - **POST**: Submit data to create a resource (e.g., form submissions).
  - **PUT**: Update existing data on the server.
  - **DELETE**: Remove data from the server.

- **Request-Response Structure**
  - **Headers**: Metadata sent with requests/responses (e.g., `Content-Type`, `User-Agent`, `Authorization`).
  - **Body**: Contains data (e.g., HTML, JSON) in responses or submitted data (e.g., form inputs) in requests.

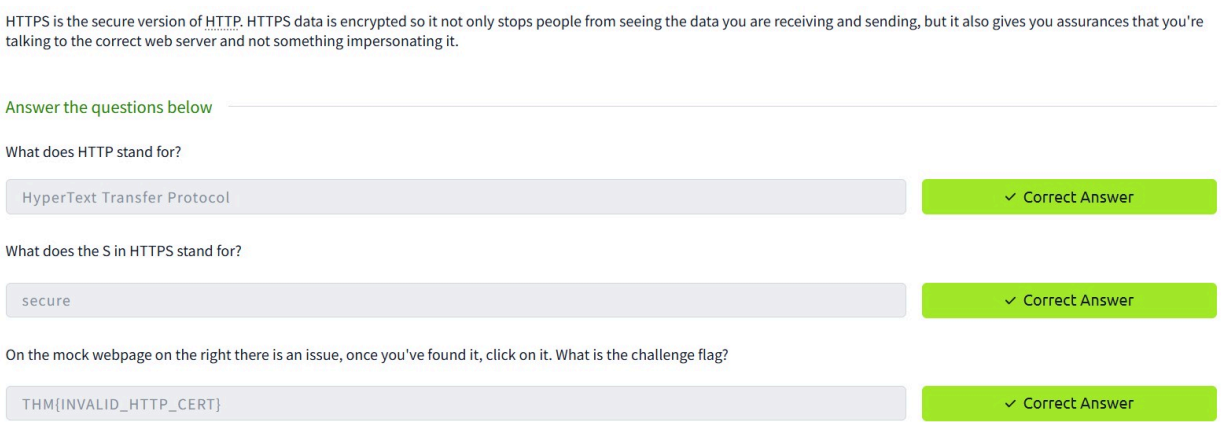
- **HTTP Status Codes**
  - **2xx (Success)**: `200 OK`, `201 Created`.

- **3xx (Redirection)**: `301 Moved Permanently`.
- **4xx (Client Errors)**: `400 Bad Request`, `401 Unauthorized`, `404 Not Found`.
- **5xx (Server Errors)**: `500 Internal Server Error`.

- **Headers Explained**
  - **Request Headers**: `Cookie`, `Accept-Language`, `Content-Type`, `Authorization`.
  - **Response Headers**: `Set-Cookie`, `Content-Type`, `Server`.

- **Tools & Demonstrations**
  - **Postman**: Test APIs by sending requests (GET, POST, etc.) and inspecting headers/body.
  - **Express.js**: Minimal Node.js framework to handle HTTP directly (e.g., routing, status codes, headers).
    - Example: Creating endpoints for GET/POST requests, validating tokens in headers, and sending JSON responses.

Tryhackme room :



Line 4: The Content-Type header tells the client what sort of information is going to be sent, such as HTML, images, videos, pdf, XML.

Line 5: Content-Length tells the client how long the response is, this way we can confirm no data is missing.

Line 6: HTTP response contains a blank line to confirm the end of the HTTP response.

Lines 7-14: The information that has been requested, in this instance the homepage.

Answer the questions below

What HTTP protocol is being used in the above example?

HTTP/1.1 ✓ Correct Answer

What response header tells the browser how much data to expect?

Content-Length ✓ Correct Answer

sk 3 HTTP Methods

Answer the questions below

What method would be used to create a new user account?

POST ✓ Correct Answer

What method would be used to update your email address?

PUT ✓ Correct Answer

What method would be used to remove a picture you've uploaded to your account?

DELETE ✓ Correct Answer

What method would be used to view a news article?

GET ✓ Correct Answer

Room completed (100%)

Answer the questions below

What response code might you receive if you've created a new user or blog post article?

201 ✓ Correct Answer

What response code might you receive if you've tried to access a page that doesn't exist?

404 ✓ Correct Answer

What response code might you receive if the web server cannot access its database and the application crashes?

503 ✓ Correct Answer

What response code might you receive if you try to edit your profile without logging in first?

401 ✓ Correct Answer

Content-Encoding: What method has been used to compress the data to make it smaller when sending it over the internet.

Answer the questions below

What header tells the web server what browser is being used?

User-Agent ✓ Correct Answer

What header tells the browser what type of data is being returned?

Content-Type ✓ Correct Answer

What header tells the web server which website is being requested?

Host ✓ Correct Answer

Task 6 Cookies

Once you have developer tools open, click on the "Network" tab. This tab will show you a list of all the resources your browser has requested. You can click on each one to receive a detailed breakdown of the request and response. If your browser sent a cookie, you will see these on the "Cookies" tab of the request.

Answer the questions below

Which header is used to save cookies to your computer?

Set-Cookie ✓ Correct Answer

This is an emulator for making demo HTTP requests, using what you've learnt from the above tasks you can use it to complete the below questions.

Answer the questions below

Make a GET request to /room

THM{YOU'RE_IN_THE_ROOM} ✓ Correct Answer Hint

Make a GET request to /blog and using the gear icon set the id parameter to 1 in the URL field

THM{YOU_FOUND_THE_BLOG} ✓ Correct Answer

Make a DELETE request to /user/1

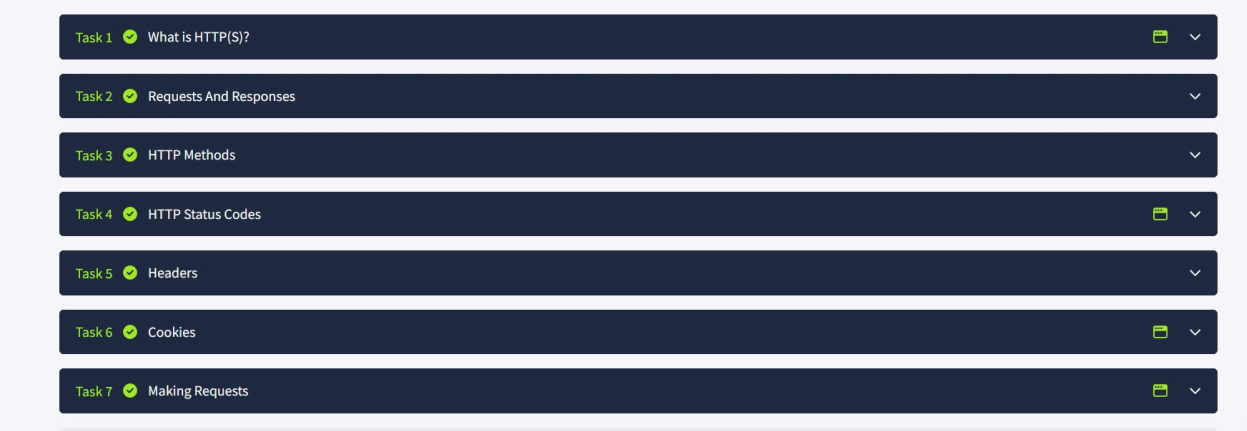
THM{USER_IS_DELETED} ✓ Correct Answer

Make a PUT request to /user/2 with the username parameter set to admin

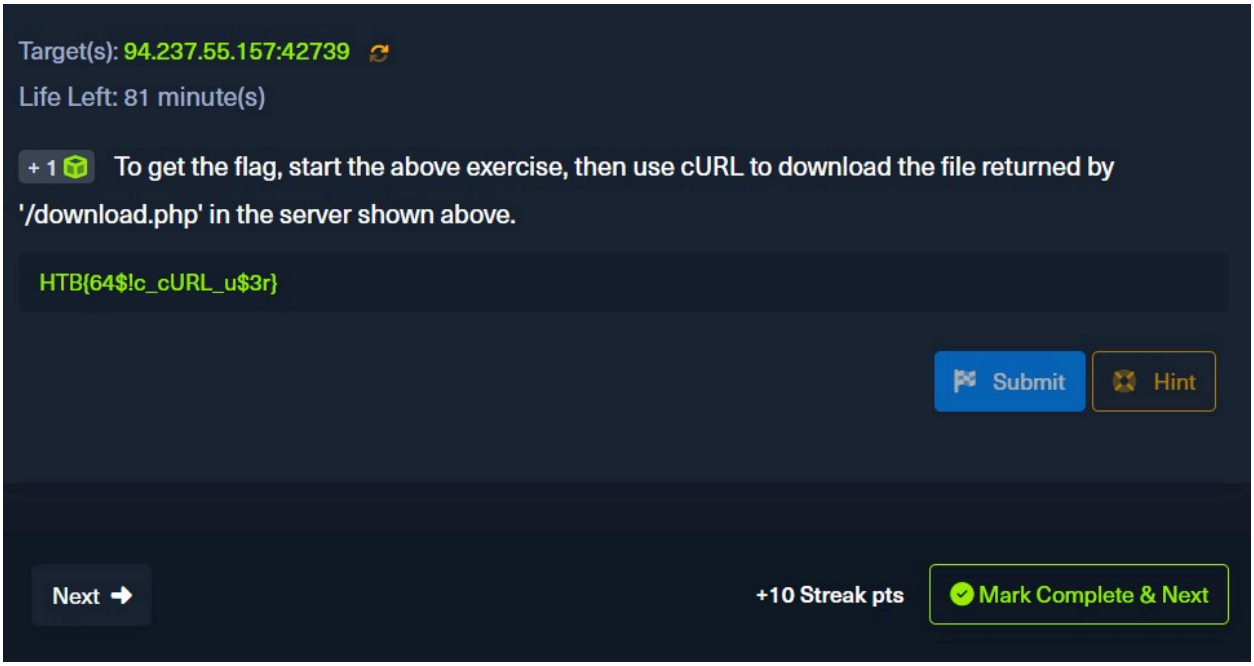
THM{USER_HAS_UPDATED} ✓ Correct Answer Hint

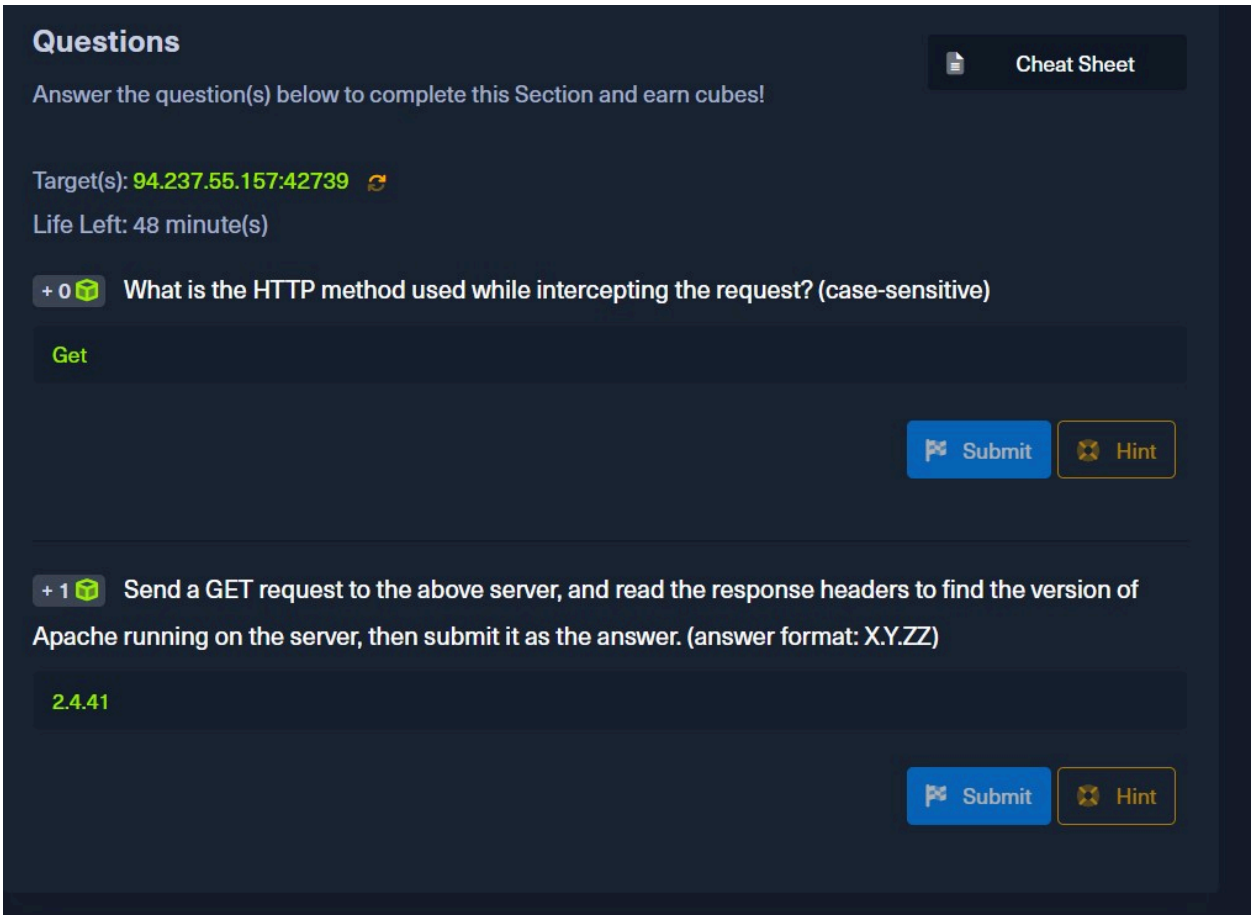
POST the username of thm and a password of letmein to /login

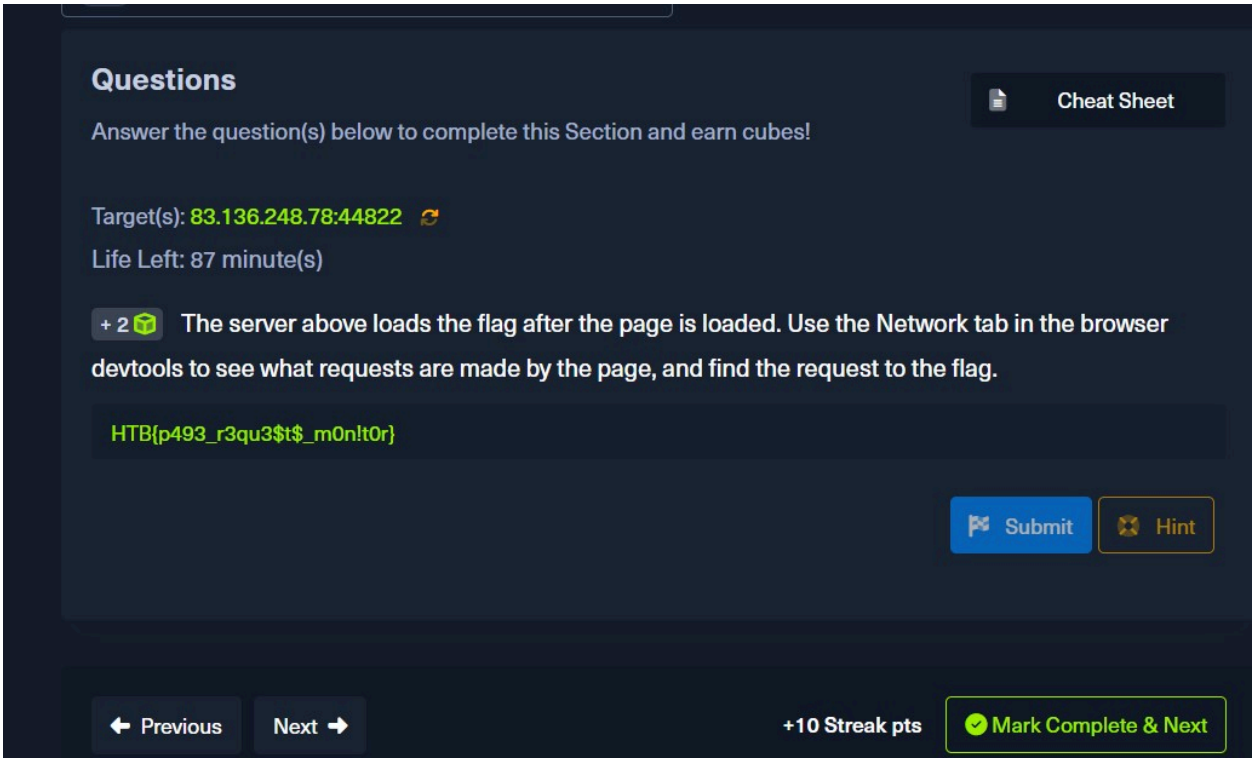
THM{HTTP_REQUEST_MASTER} ✓ Correct Answer Hint

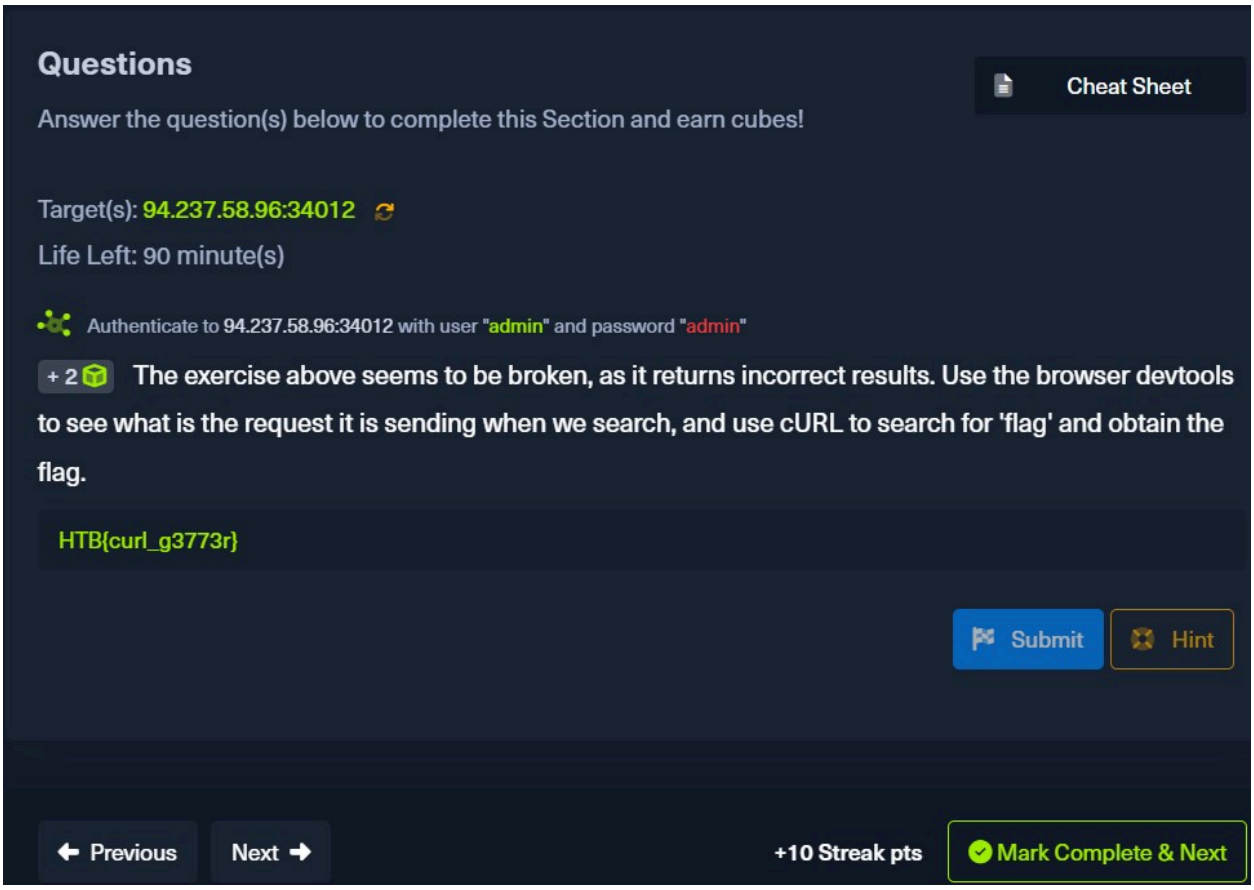

Task 1 What is HTTP(S)?
Task 2 Requests And Responses
Task 3 HTTP Methods
Task 4 HTTP Status Codes
Task 5 Headers
Task 6 Cookies
Task 7 Making Requests

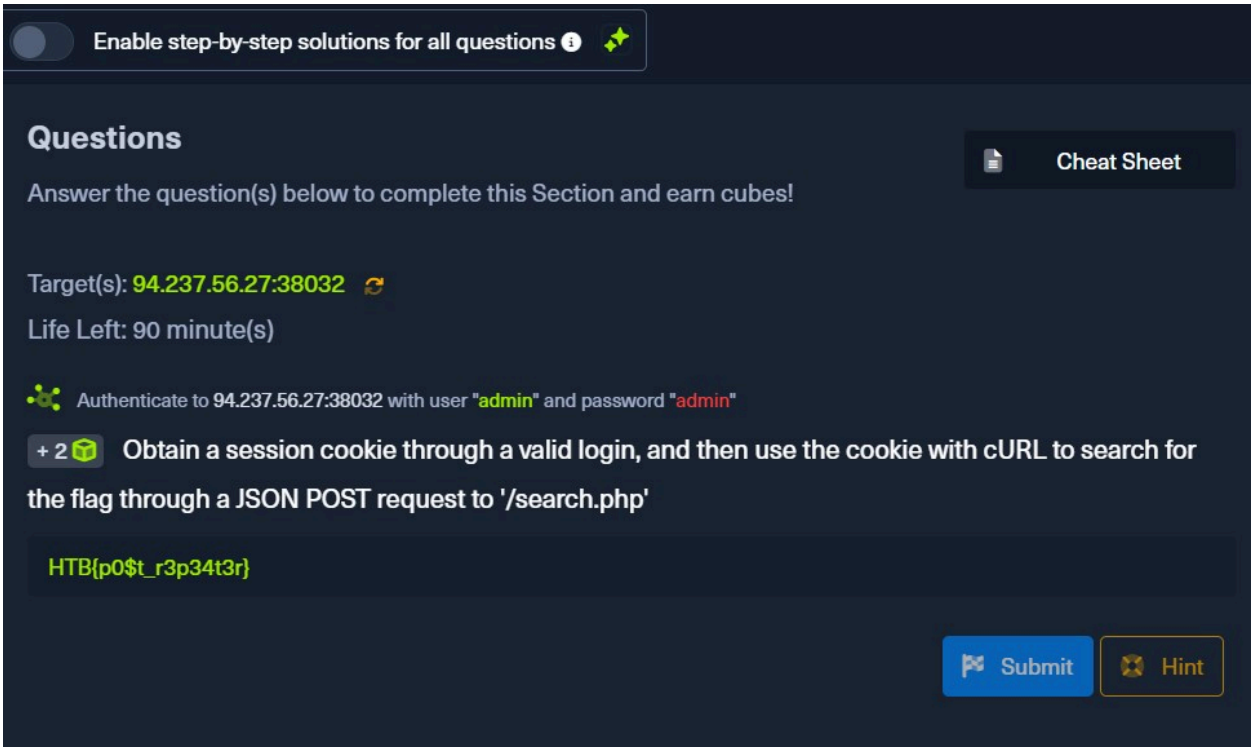
THB room :

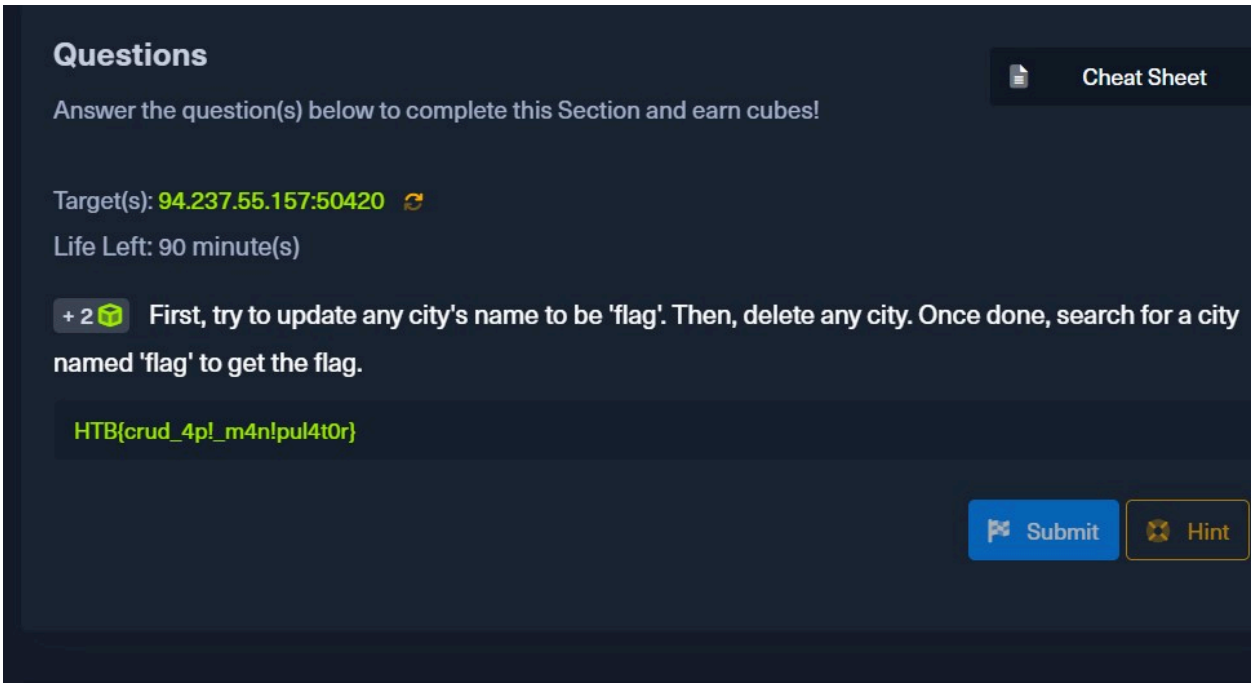

Target(s): 94.237.55.157:42739
Life Left: 81 minute(s)
+ 1
To get the flag, start the above exercise, then use cURL to download the file returned by '/download.php' in the server shown above.
HTB{64$!c_cURL_u$3r}
Submit
Hint
Next
+10 Streak pts
Mark Complete & Next


Questions
Cheat Sheet
Answer the question(s) below to complete this Section and earn cubes!
Target(s): 94.237.55.157:42739
Life Left: 48 minute(s)
+ 0
What is the HTTP method used while intercepting the request? (case-sensitive)
Get
Submit
Hint
+ 1
Send a GET request to the above server, and read the response headers to find the version of Apache running on the server, then submit it as the answer. (answer format: X.Y.ZZ)
2.4.41
Submit
Hint


Questions
Cheat Sheet
Answer the question(s) below to complete this Section and earn cubes!
Target(s): 83.136.248.78:44822
Life Left: 87 minute(s)
+ 2
The server above loads the flag after the page is loaded. Use the Network tab in the browser devtools to see what requests are made by the page, and find the request to the flag.
HTB{p493_r3qu3t_m0n!t0r}
Submit
Hint
Previous
Next
+10 Streak pts
Mark Complete & Next


Questions
Cheat Sheet
Answer the question(s) below to complete this Section and earn cubes!
Target(s): 94.237.58.96:34012
Life Left: 90 minute(s)
Authenticate to 94.237.58.96:34012 with user "admin" and password "admin"
+ 2
The exercise above seems to be broken, as it returns incorrect results. Use the browser devtools to see what is the request it is sending when we search, and use cURL to search for 'flag' and obtain the flag.
HTB{curl_g3773r}
Submit
Hint
Previous
Next
+10 Streak pts
Mark Complete & Next


Enable step-by-step solutions for all questions
Questions
Cheat Sheet
Answer the question(s) below to complete this Section and earn cubes!
Target(s): 94.237.56.27:38032
Life Left: 90 minute(s)
Authenticate to 94.237.56.27:38032 with user "admin" and password "admin"
+ 2
Obtain a session cookie through a valid login, and then use the cookie with cURL to search for the flag through a JSON POST request to '/search.php'
HTB{p0$t_r3p34t3r}
Submit
Hint


Questions
Cheat Sheet
Answer the question(s) below to complete this Section and earn cubes!
Target(s): 94.237.55.157:50420
Life Left: 90 minute(s)
+ 2
First, try to update any city's name to be 'flag'. Then, delete any city. Once done, search for a city named 'flag' to get the flag.
HTB{crud_4p!_m4n!pul4t0r}
Submit
Hint