Cloud and Network Security-C1-2026

Student Name: **Felix Webbo**

Student No: **CS-CNS11-26044**

SUNDAY, FEB 01, 2026

Week 3: Assignment 2

Class Exercise: **HTB Academy - Web Requests**

1 ABSTRACT

This report documented the completion of the *Web Requests* module on Hack The Box Academy. The objective of the module was to develop a foundational understanding of how web applications communicate using the HyperText Transfer Protocol (HTTP) and Hypertext Transfer Protocol Secure (HTTPS). The study examined HTTP request and response structures, headers, methods, status codes, and basic interaction with APIs using tools such as cURL and browser Developer Tools. The module reinforced essential skills required for web application security testing and traffic analysis. The report presented the methodology, key findings, and reflective learning outcomes.

Table of Contents

2 INTRODUCTION

Web applications relied heavily on HTTP and HTTPS for client–server communication. Understanding how web requests were structured, transmitted, and processed was essential for both web development and cybersecurity operations. This assignment involved completing the Web Requests module on Hack The Box Academy to gain practical and theoretical knowledge of web communication mechanisms. The module focused on HTTP fundamentals, secure communication, request methods, response codes, and API interaction. The aim was to strengthen skills relevant to web application testing, penetration testing, and secure system design.

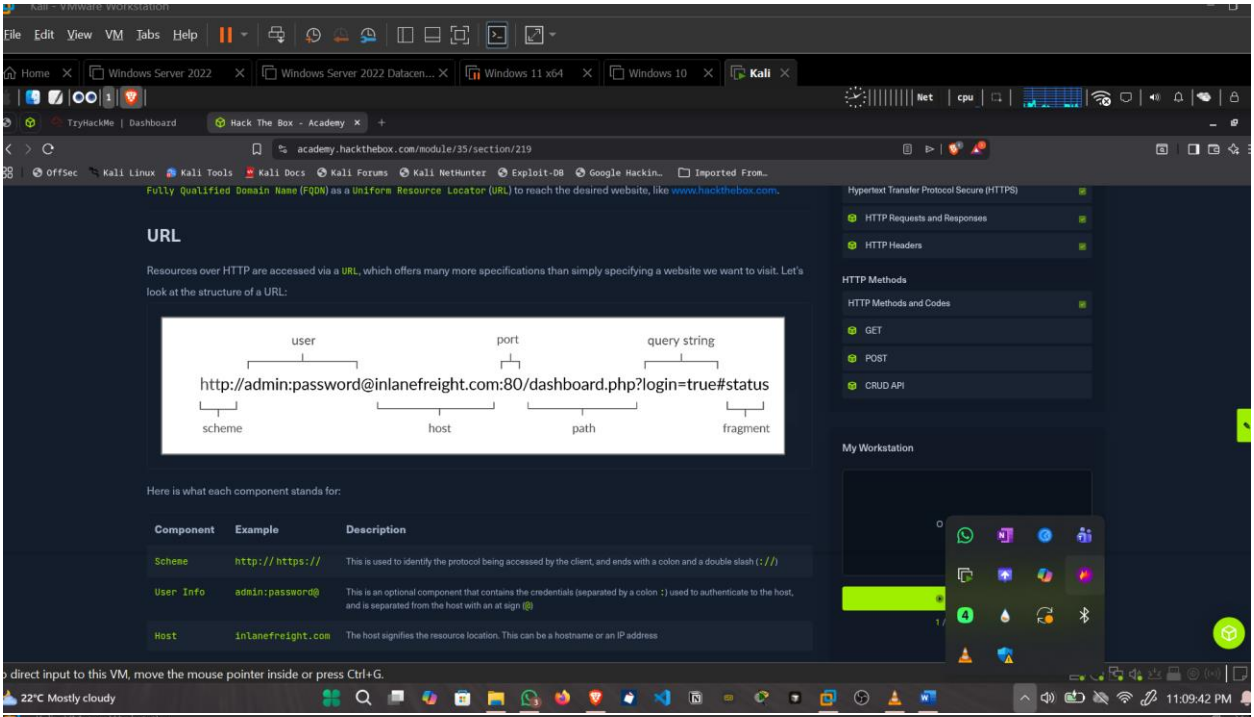
3 METHODOLOGY

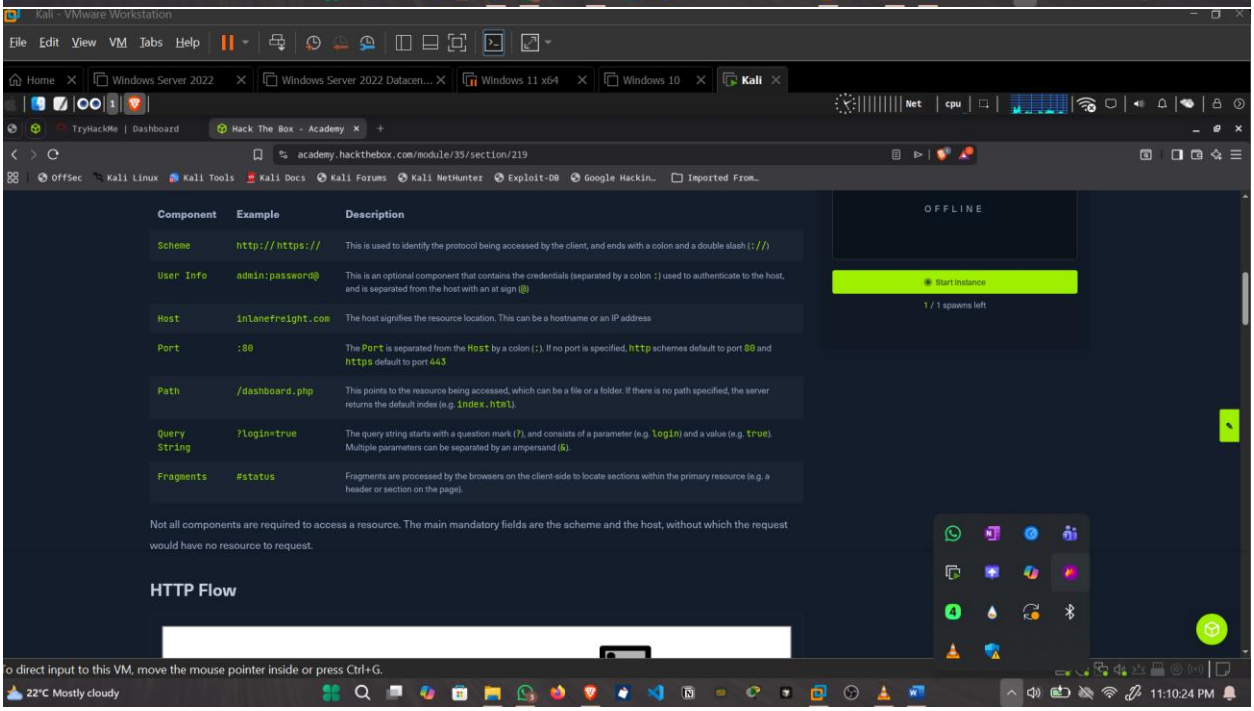
The module was completed using the Hack The Box Academy learning platform. Each section was studied sequentially, combining conceptual explanations with hands-on demonstrations using cURL and browser Developer Tools. Embedded questions and exercises were completed based on observed request and response behavior. Screenshots were captured to document task completion and learning evidence. Upon completion, a shareable link and final completion screenshot were generated and published on social media to demonstrate professional development.

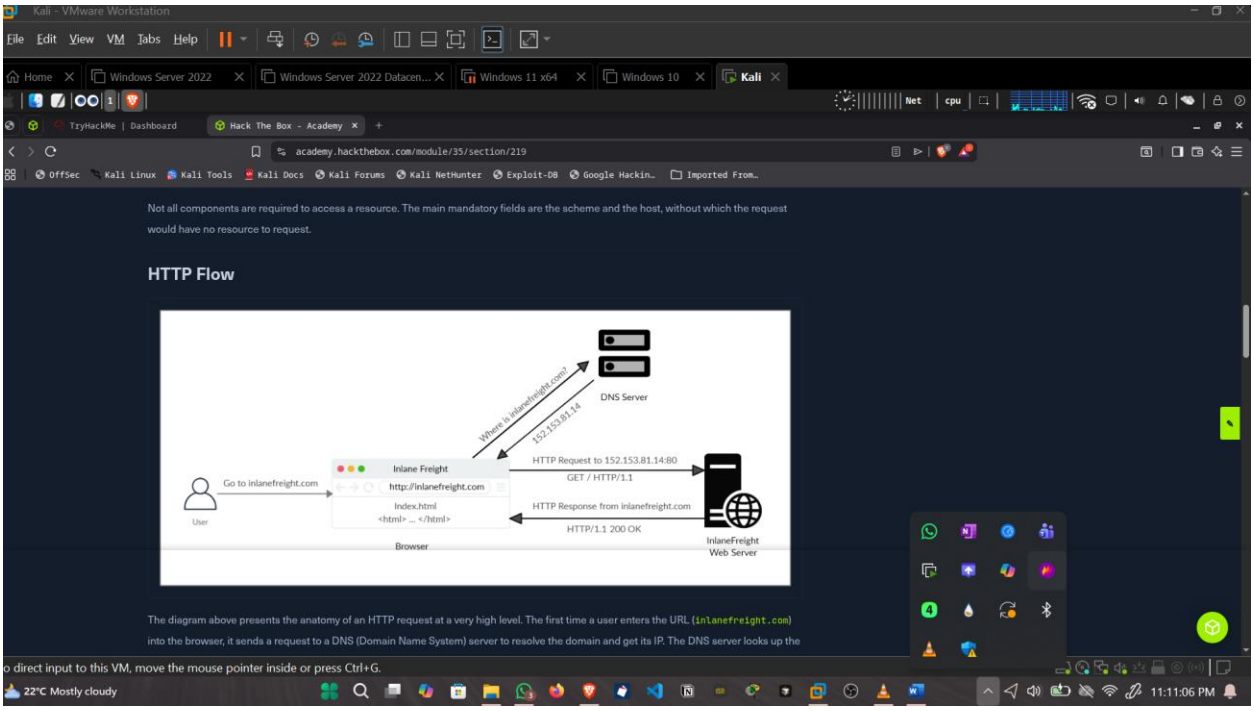
4 OVERVIEW OF WEB REQUESTS

4.1 HyperText Transfer Protocol (HTTP)

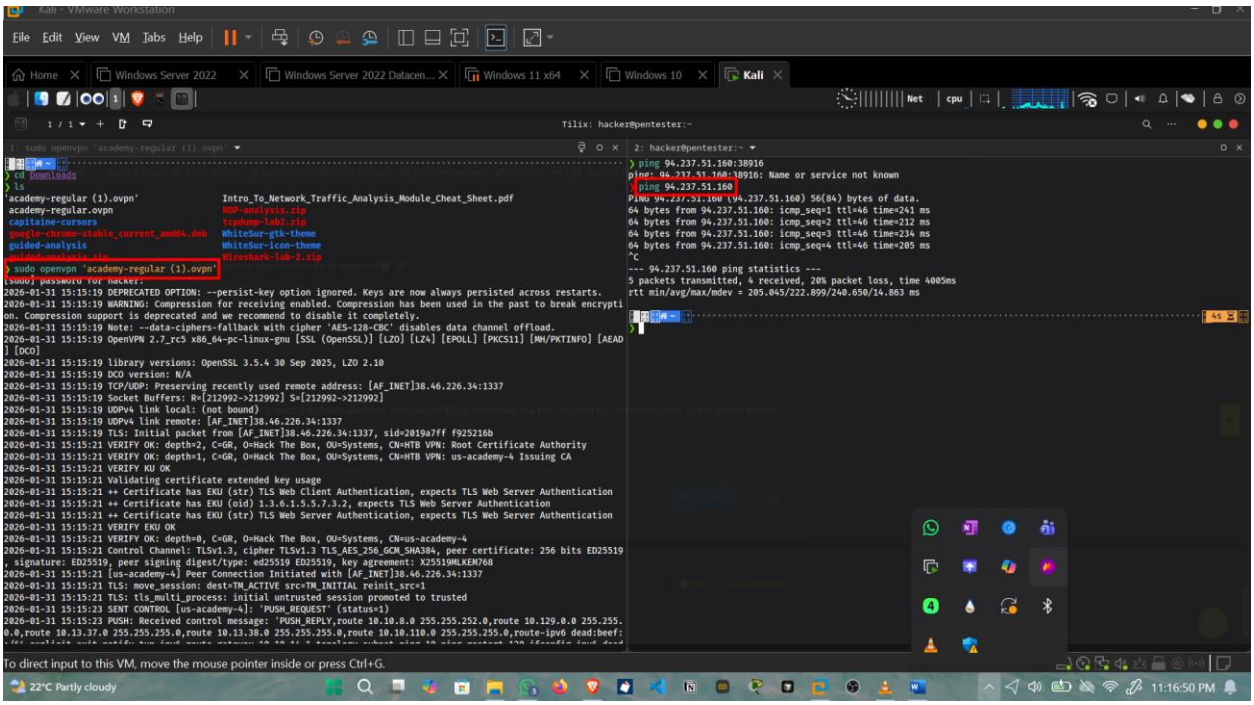
HTTP was identified as a stateless application-layer protocol used to transmit data between clients and servers. Communication occurred through structured requests and responses, enabling browsers to retrieve web content. However, HTTP traffic was transmitted in plaintext, making it vulnerable to interception and manipulation.

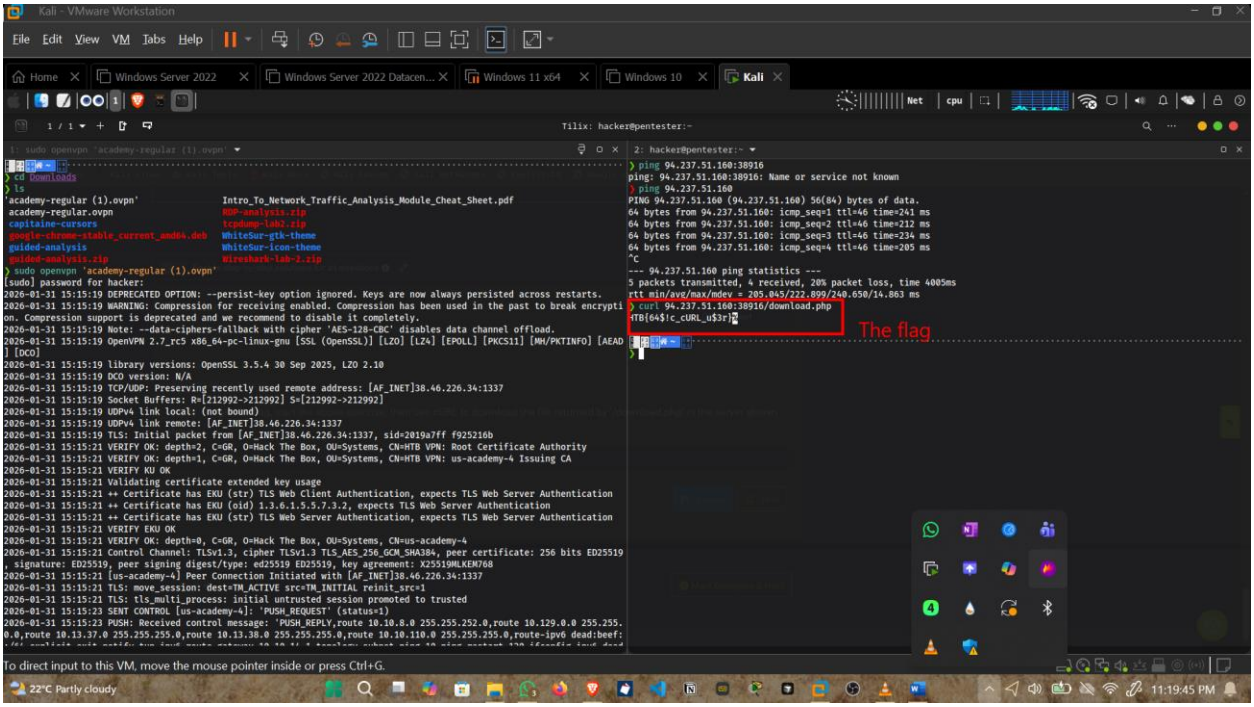

Fully Qualified Domain Name (FQDN) as a Uniform Resource Locator (URL) to reach the desired website, like www.hackthebox.com.
URL
Resources over HTTP are accessed via a URL, which offers many more specifications than simply specifying a website we want to visit. Let's look at the structure of a URL:
user
port
query string
http://admin:password@inlanefreight.com:80/dashboard.php?login=true#status
scheme
host
path
fragment
Here is what each component stands for:
Component
Example
Description
Scheme
http:// https://
This is used to identify the protocol being accessed by the client, and ends with a colon and a double slash (://)
User Info
admin:password@
This is an optional component that contains the credentials (separated by a colon :) used to authenticate to the host, and is separated from the host with an at sign (@)
Host
inlanefreight.com
The host signifies the resource location. This can be a hostname or an IP address
Hypertext Transfer Protocol Secure (HTTPS)
HTTP Requests and Responses
HTTP Headers
HTTP Methods
HTTP Methods and Codes
GET
POST
CRUD API
My Workstation
22°C Mostly cloudy
11:09:42 PM


Component
Example
Description
Scheme
http:// https://
This is used to identify the protocol being accessed by the client, and ends with a colon and a double slash (://)
User Info
admin:password@
This is an optional component that contains the credentials (separated by a colon :) used to authenticate to the host, and is separated from the host with an at sign (@)
Host
inlanefreight.com
The host signifies the resource location. This can be a hostname or an IP address
Port
:80
The Port is separated from the Host by a colon (:). If no port is specified, http schemes default to port 80 and https default to port 443
Path
/dashboard.php
This points to the resource being accessed, which can be a file or a folder. If there is no path specified, the server returns the default index (e.g. index.html).
Query String
?login=true
The query string starts with a question mark (?), and consists of a parameter (e.g. login) and a value (e.g. true). Multiple parameters can be separated by an ampersand (&).
Fragments
#status
Fragments are processed by the browsers on the client-side to locate sections within the primary resource (e.g. a header or section on the page).
OFFLINE
Start Instance
1 / 1 spawns left
Not all components are required to access a resource. The main mandatory fields are the scheme and the host, without which the request would have no resource to request.
HTTP Flow
22°C Mostly cloudy
11:10:24 PM



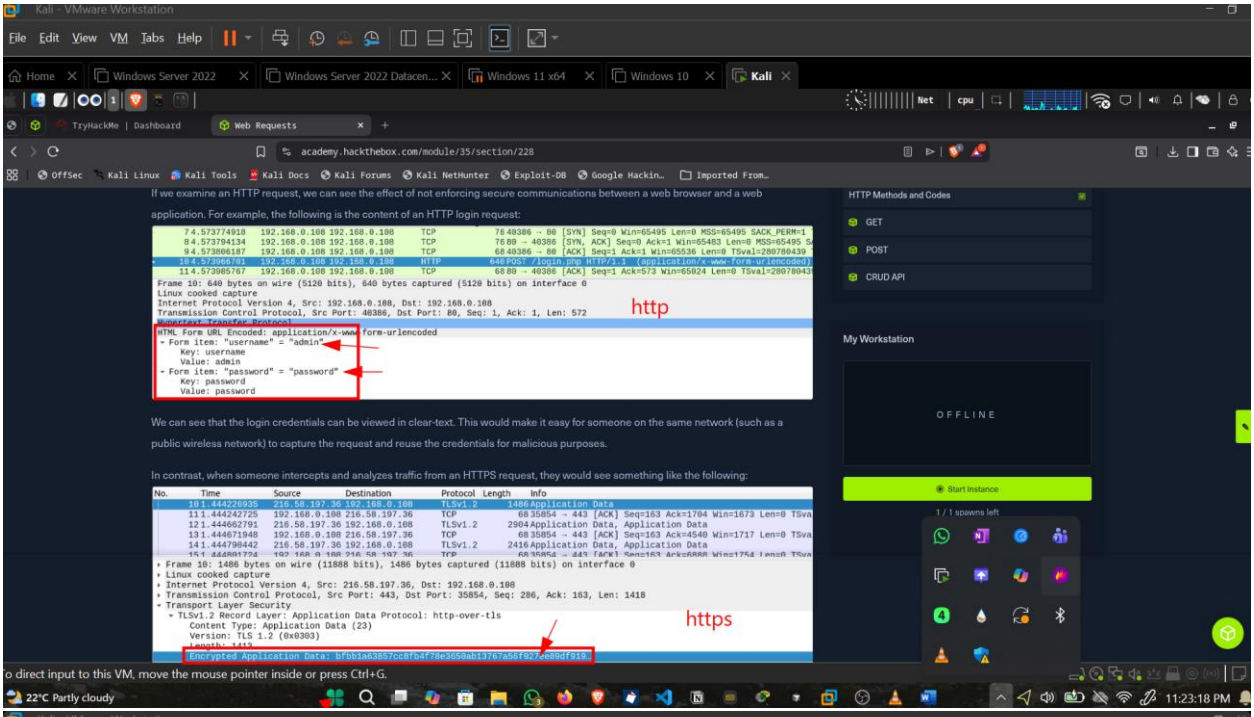
Connecting via vpn

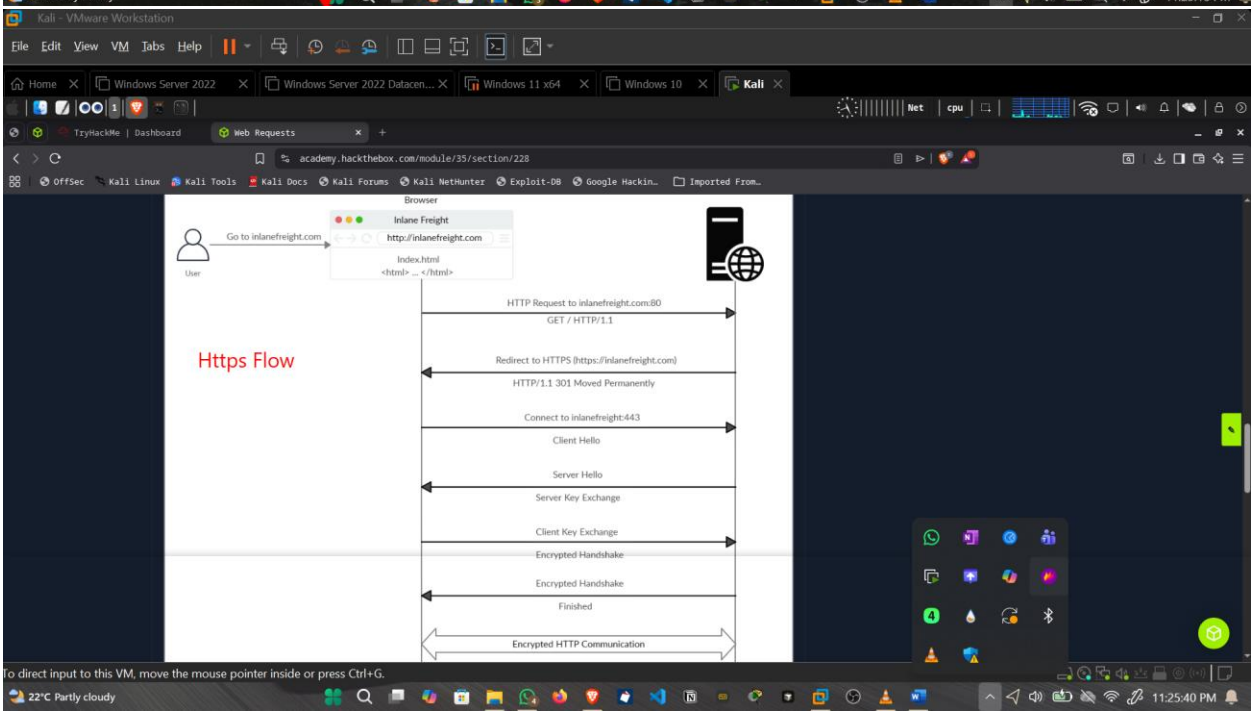




4.2 Hypertext Transfer Protocol Secure (HTTPS)

HTTPS extended HTTP by incorporating encryption through Transport Layer Security (TLS). This ensured confidentiality, integrity, and authentication between communicating parties. HTTPS was shown to be critical in protecting sensitive data such as credentials and session tokens from interception.

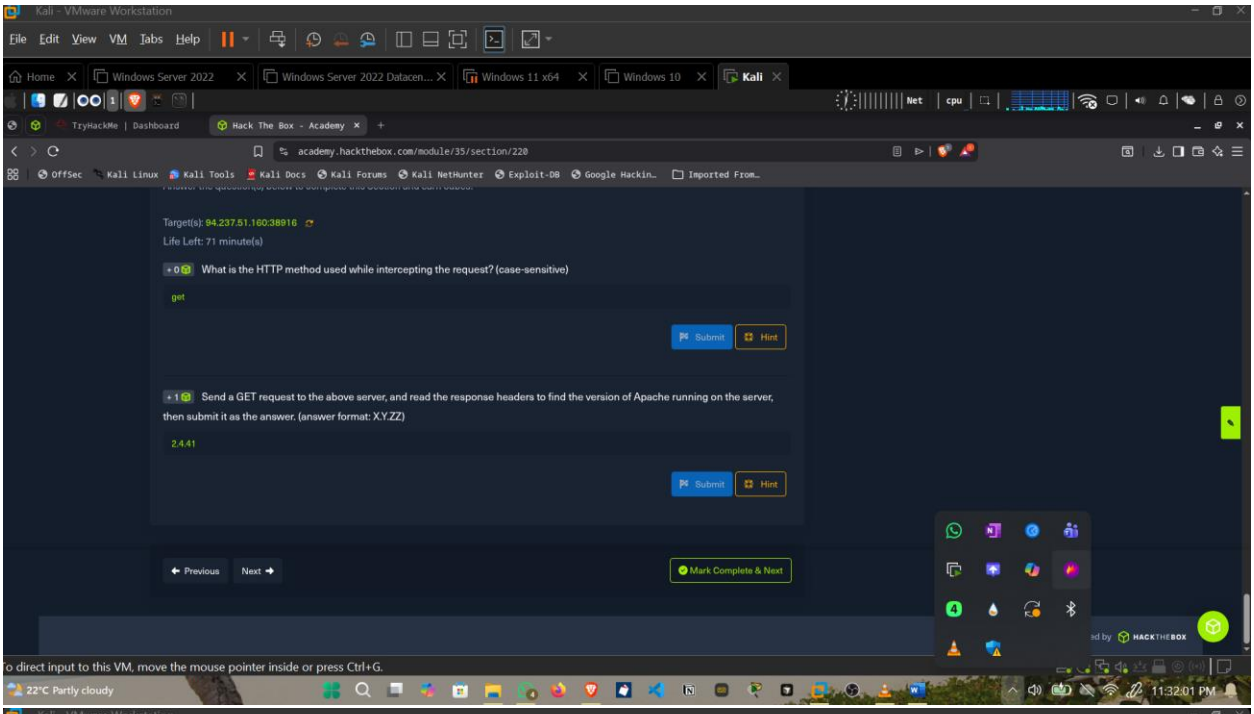


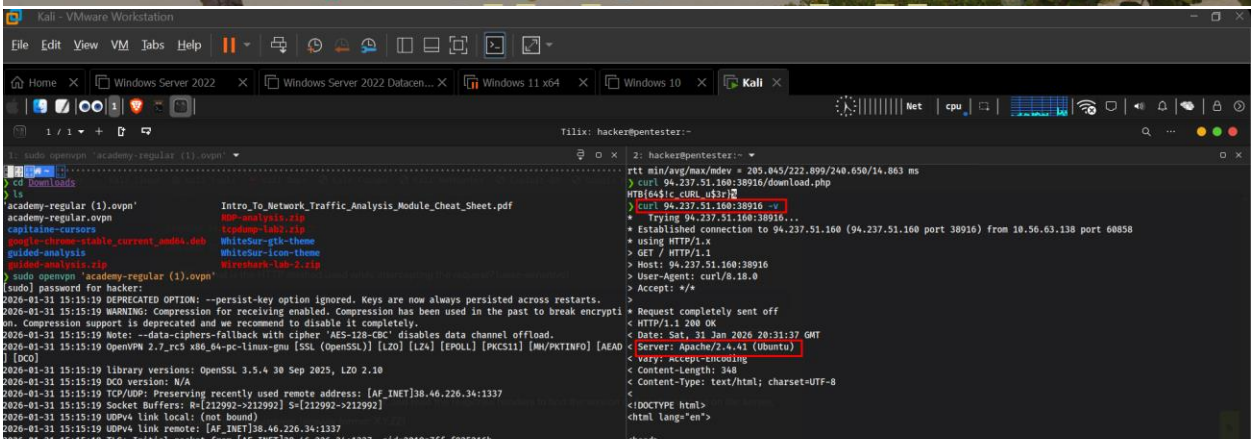


4.3 HTTP Requests and Responses

HTTP communication consisted of requests sent by clients and responses returned by servers. Each request and response contained headers and, optionally, a body. Headers carried metadata such as

content type, authorization details, cookies, and user-agent information. Understanding these components was essential for analyzing web traffic and identifying vulnerabilities.

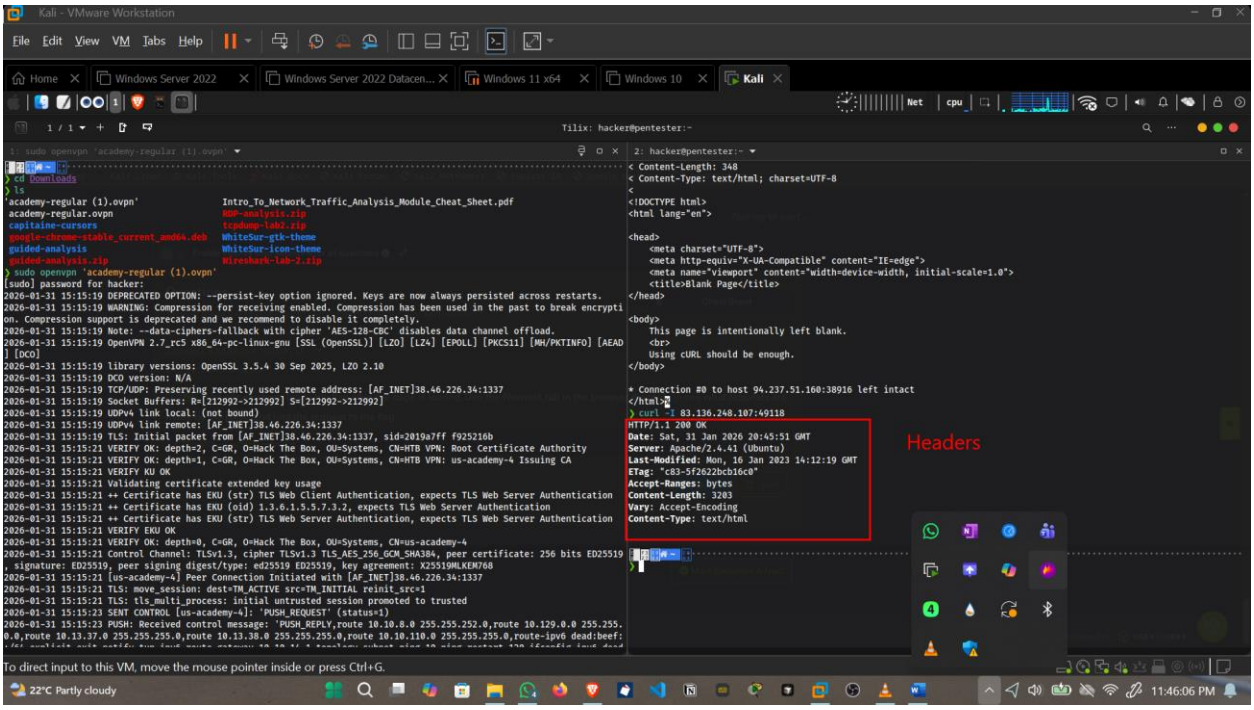


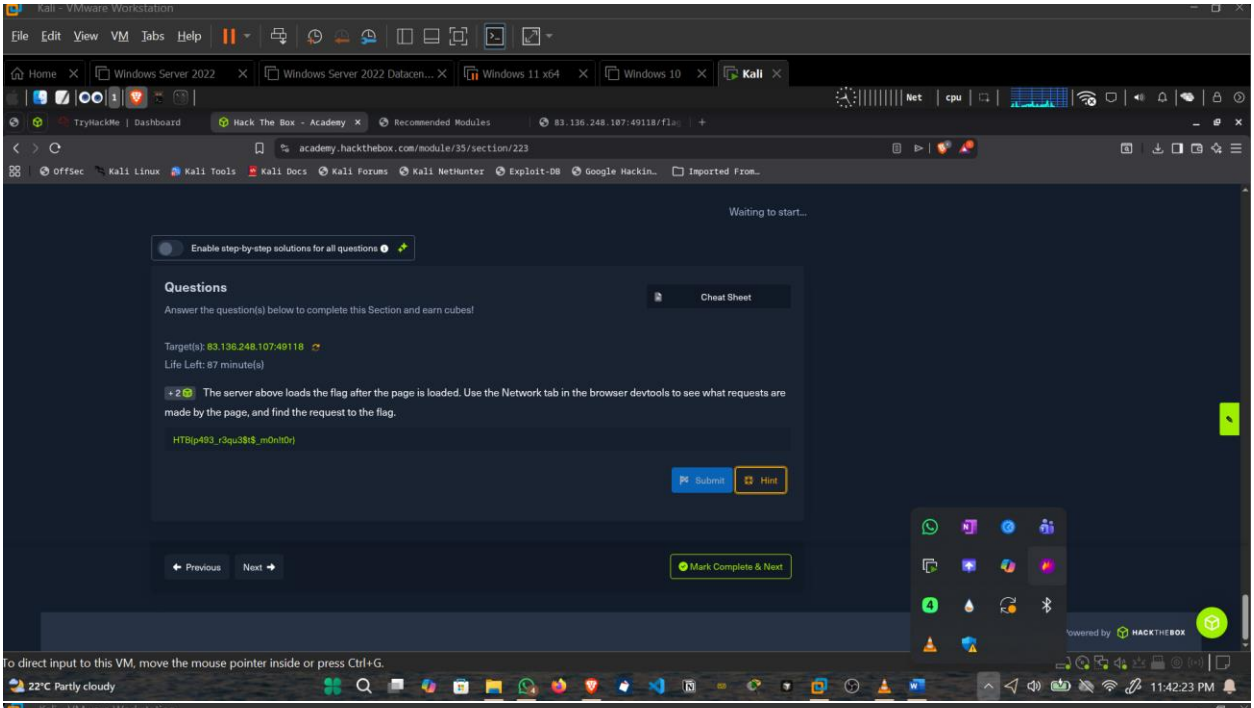


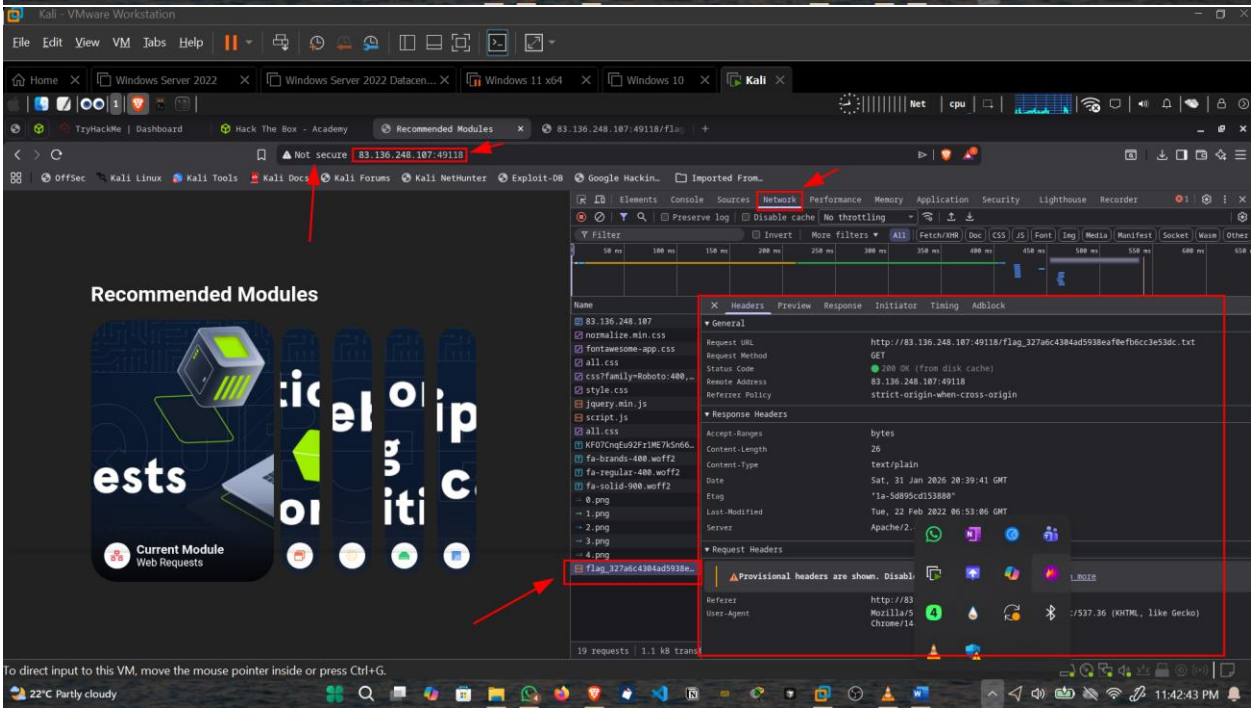
4.4 HTTP Headers

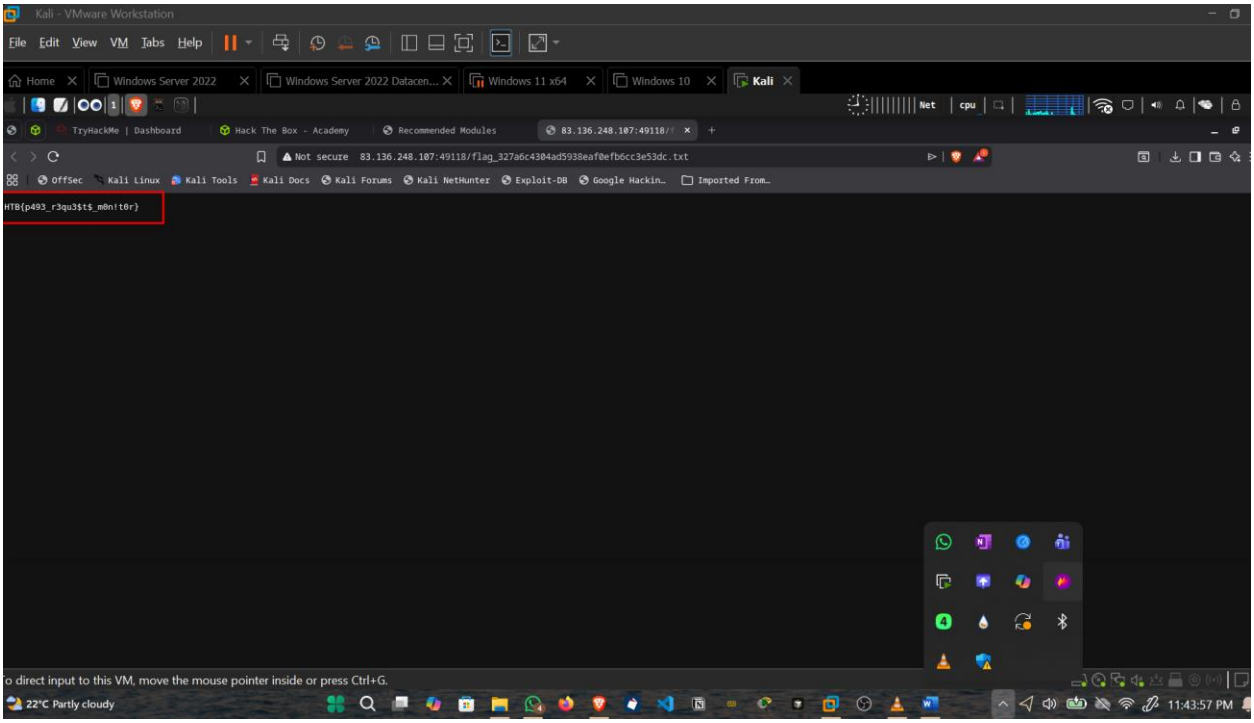
HTTP headers were used to exchange metadata between the client and the server during web communication. These headers provided contextual information about requests and responses, such as content type, authentication details, and caching behavior. Some headers were specific to requests or responses, while others were shared across both.

Headers contained one or more values appended after the header name and separated by a colon. HTTP headers were categorized into **General Headers**, **Entity Headers**, **Request Headers**, **Response Headers**, and **Security Headers**, each serving a distinct role in controlling how data was transmitted, processed, and protected.



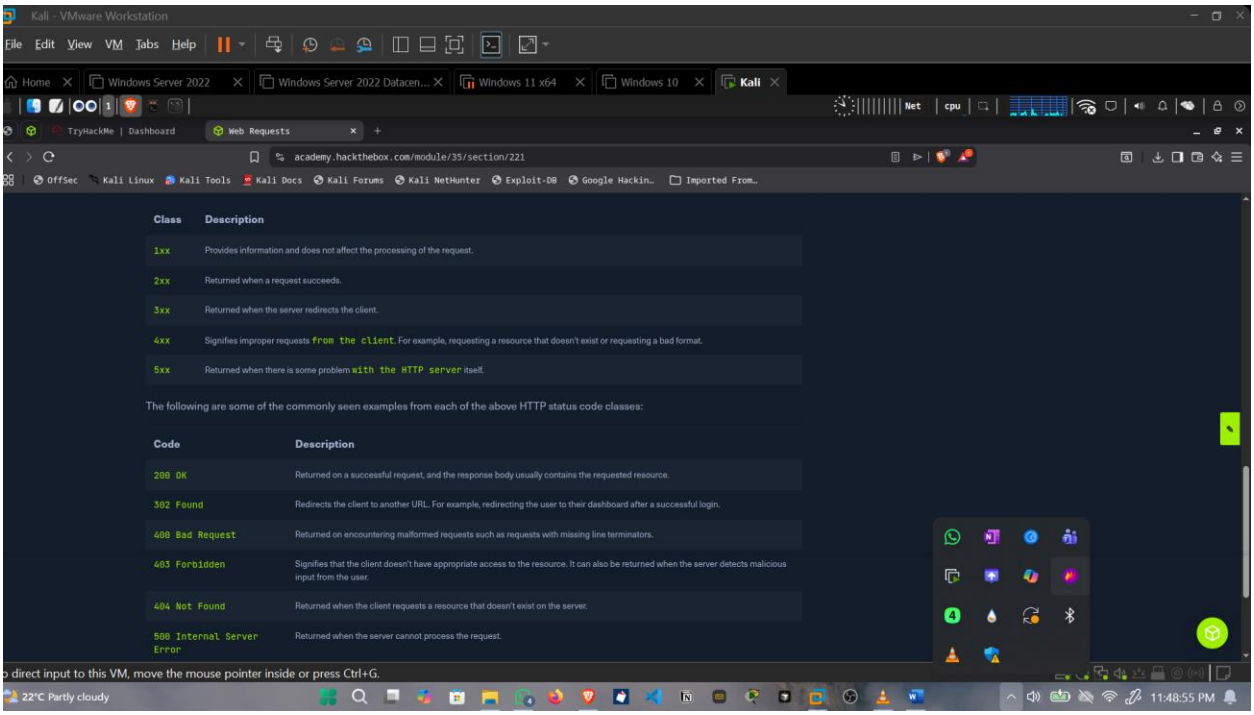

Kali - VMware Workstation
File Edit View VM Tabs Help
Home
Windows Server 2022
Windows Server 2022 Datacen...
Windows 11 x64
Windows 10
Kali
Hack The Box - Academy
Recommended Modules
academy.hackthebox.com/module/35/section/223
Waiting to start...
Enable step-by-step solutions for all questions
Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
Target(s): 83.136.248.107:49118
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_mOnlt0r}
Submit
Hint
Previous
Next
Mark Complete & Next
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
22°C Partly cloudy
11:42:23 PM


Kali - VMware Workstation
File Edit View VM Tabs Help
Not secure 83.136.248.107:49118
Recommended Modules
Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder
Headers Preview Response Initiator Timing Adblock
General
Request URL http://83.136.248.107:49118/flag_327a6c4304ad5938eaf0efb6cc3e53dc.txt
Request Method GET
Status Code 200 OK (from disk cache)
Remote Address 83.136.248.107:49118
Referrer Policy strict-origin-when-cross-origin
Response Headers
Accept-Ranges bytes
Content-Length 26
Content-Type text/plain
Date Sat, 31 Jan 2026 20:39:41 GMT
Etag "1a-5d895cd153880"
Last-Modified Tue, 22 Feb 2022 06:53:06 GMT
Request Headers
Recommended Modules
Current Module
Web Requests
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
22°C Partly cloudy
11:42:43 PM



4.5 HTTP Methods and Codes

The module examined common HTTP methods including **GET**, **POST**, **PUT**, and **DELETE**, each defining a specific action on server resources. HTTP response status codes were used to indicate request outcomes, such as success (200-series), redirection (300-series), client errors (400-series), and server errors (500-series). These codes were valuable for diagnosing application behavior and identifying misconfigurations.



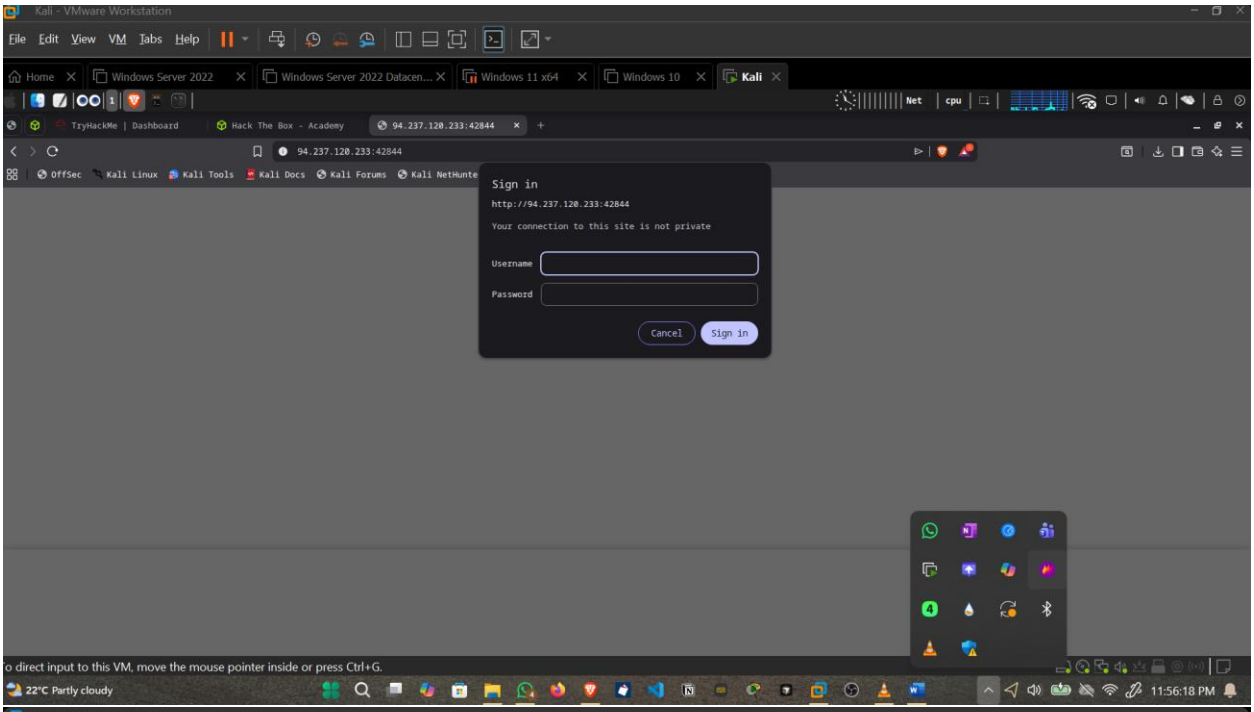
4.5.1 Get

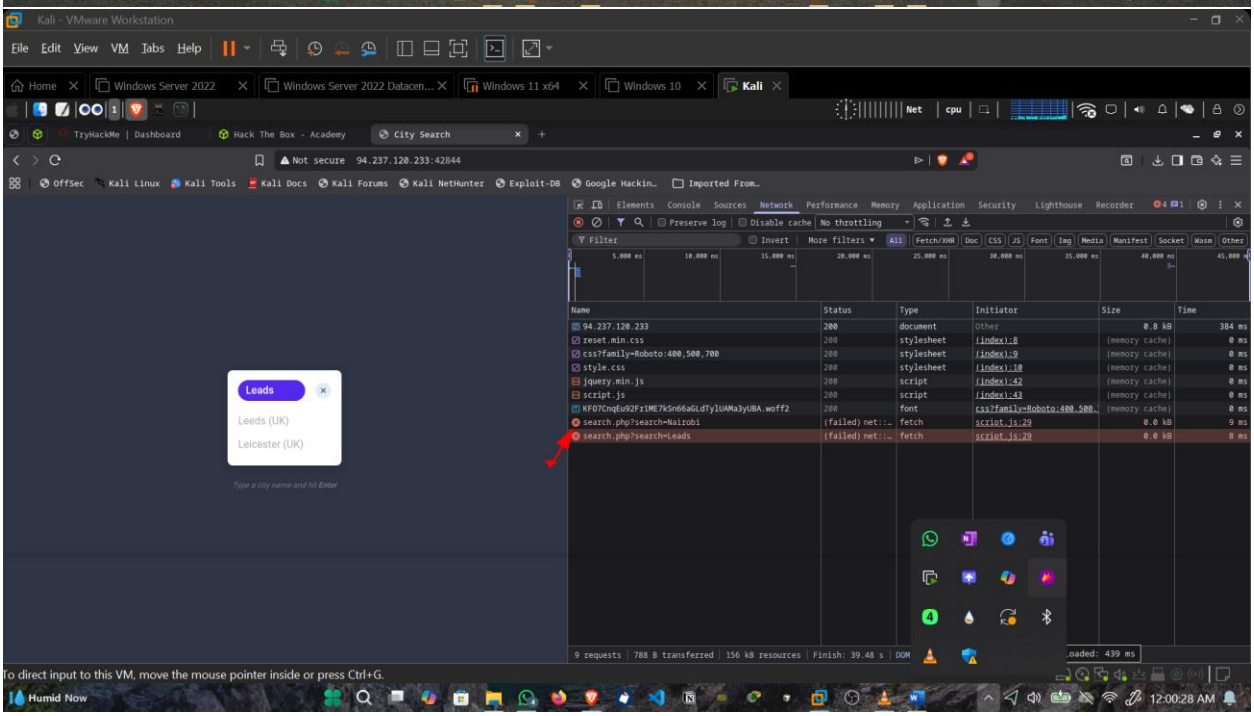
The GET method was used by web browsers to retrieve resources hosted at a specified URL. When a URL was accessed, the browser automatically issued a GET request to obtain the requested web page. After receiving the initial response, the browser often sent additional requests using various HTTP methods to load related resources such as images, stylesheets, and scripts. These requests could be observed through the Network tab in the browser's developer tools.

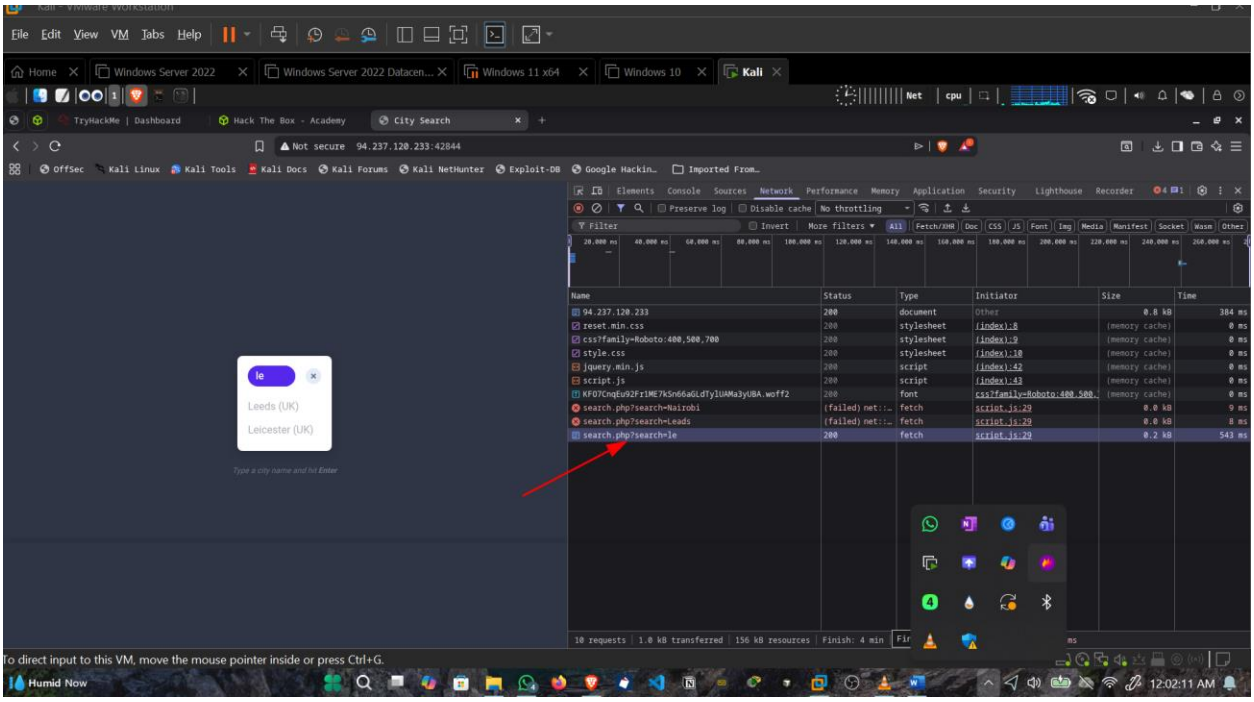
4.5.1.1 HTTP Basic Authentication

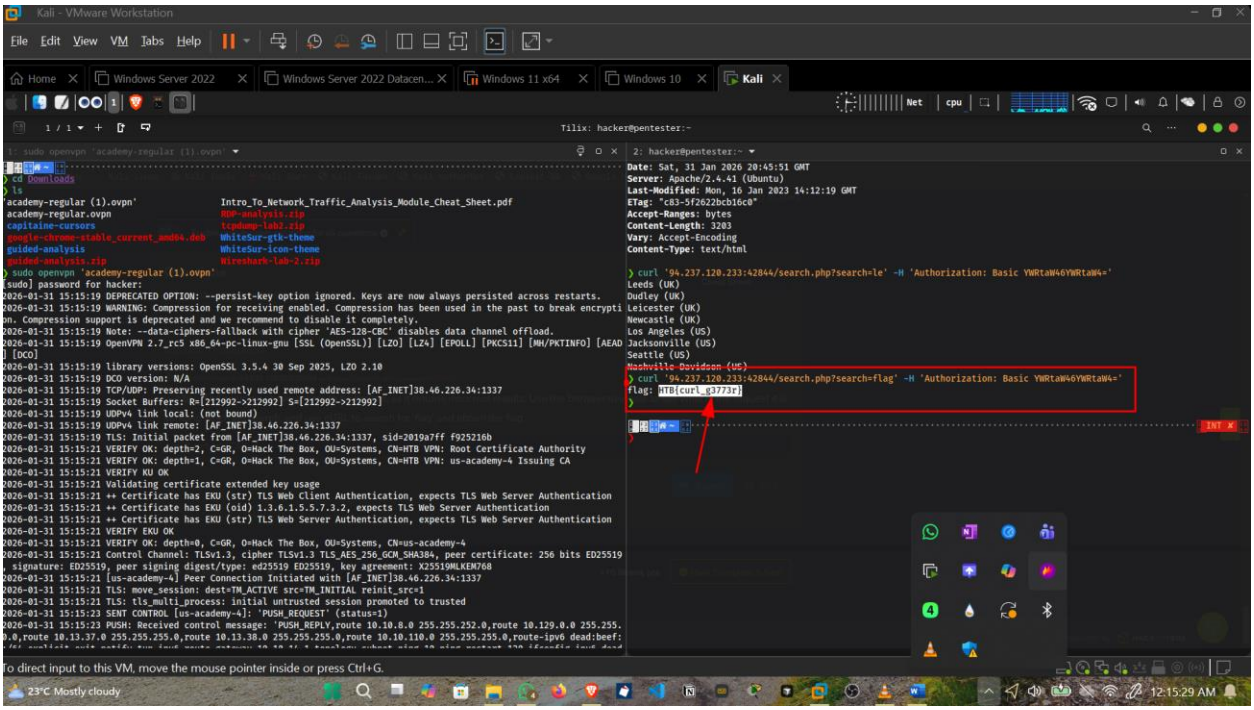
HTTP Basic Authentication was observed during the exercise at the end of the section. When accessing the protected resource, the browser prompted for a username and password before granting access. Unlike traditional login forms that transmit credentials through HTTP parameters (commonly via POST requests), Basic Authentication was handled directly by the web server to restrict access to a specific page or directory.

Access to the protected resource required valid credentials, which were provided as *admin:admin* for the purpose of the exercise. This authentication method illustrated how access control could be enforced at the server level without direct interaction with the web application logic.


Kali - VMware Workstation
File Edit View VM Tabs Help
Home
Windows Server 2022
Windows Server 2022 Datacen...
Windows 11 x64
Windows 10
Kali
TryHackMe | Dashboard
Hack The Box - Academy
94.237.120.233:42844
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Sign in
http://94.237.120.233:42844
Your connection to this site is not private
Username
Password
Cancel
Sign in
o direct input to this VM, move the mouse pointer inside or press Ctrl+G.
22°C Partly cloudy
11:56:18 PM


Kali - VMware Workstation
File Edit View VM Tabs Help
Home
Windows Server 2022
Windows Server 2022 Datacen...
Windows 11 x64
Windows 10
Kali
TryHackMe | Dashboard
Hack The Box - Academy
City Search
Not secure 94.237.120.233:42844
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hackin...
Imported From...
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Leads
Leeds (UK)
Leicester (UK)
Type a city name and hit Enter
Name
Status
Type
Initiator
Size
Time
94.237.120.233
reset.min.css
css?family=Roboto:400,500,700
style.css
jquery.min.js
script.js
search.php?search=Nairobi
search.php?search=Leads
9 requests
788 B transferred
156 kB resources
Finish: 39.48 s
DOM
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
Humid Now
12:00:28 AM

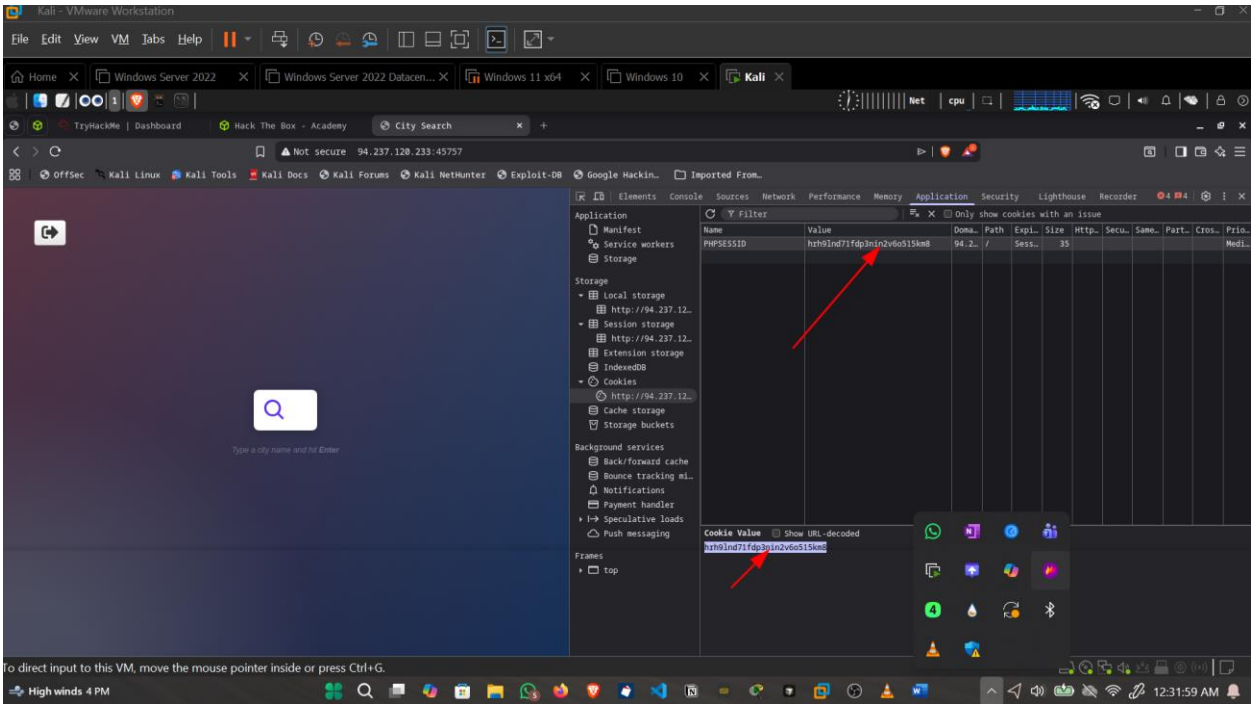


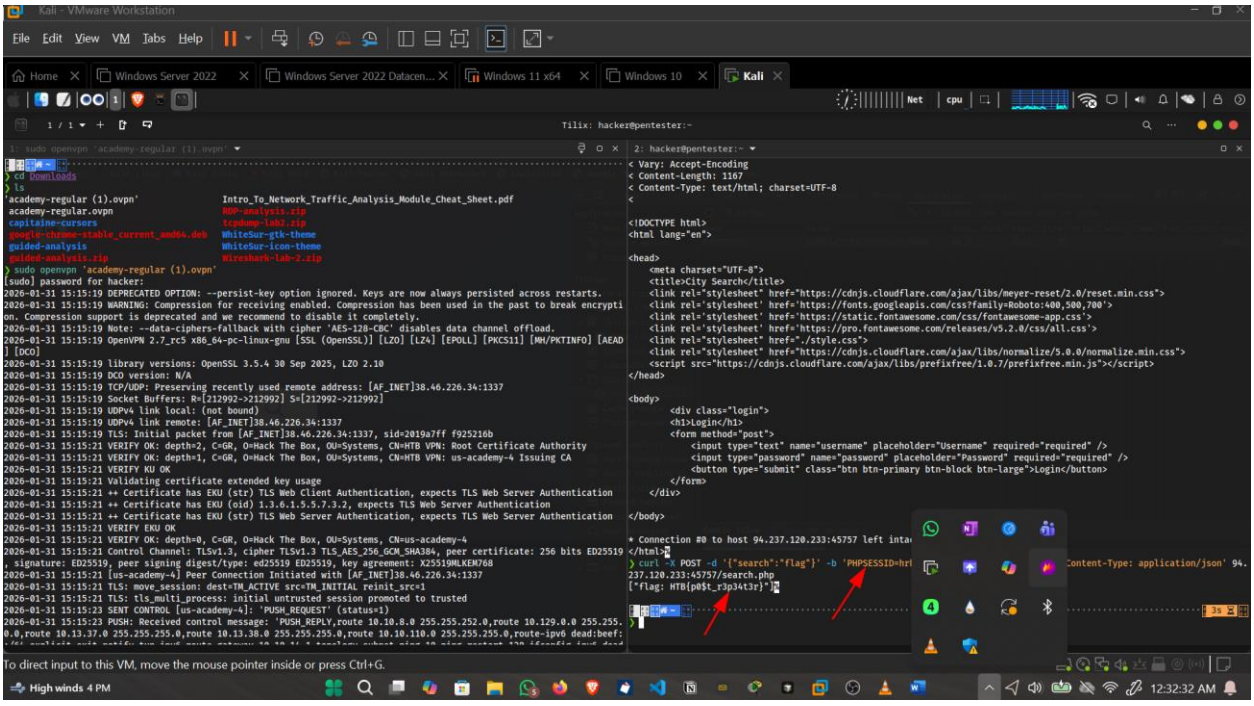


4.5.2 POST Method

POST requests were used to send data in the HTTP request body instead of the URL. This allowed secure transfer of larger data, reduced logging of sensitive information, and required less encoding. During the exercise, POST was observed in a PHP login form, demonstrating how web

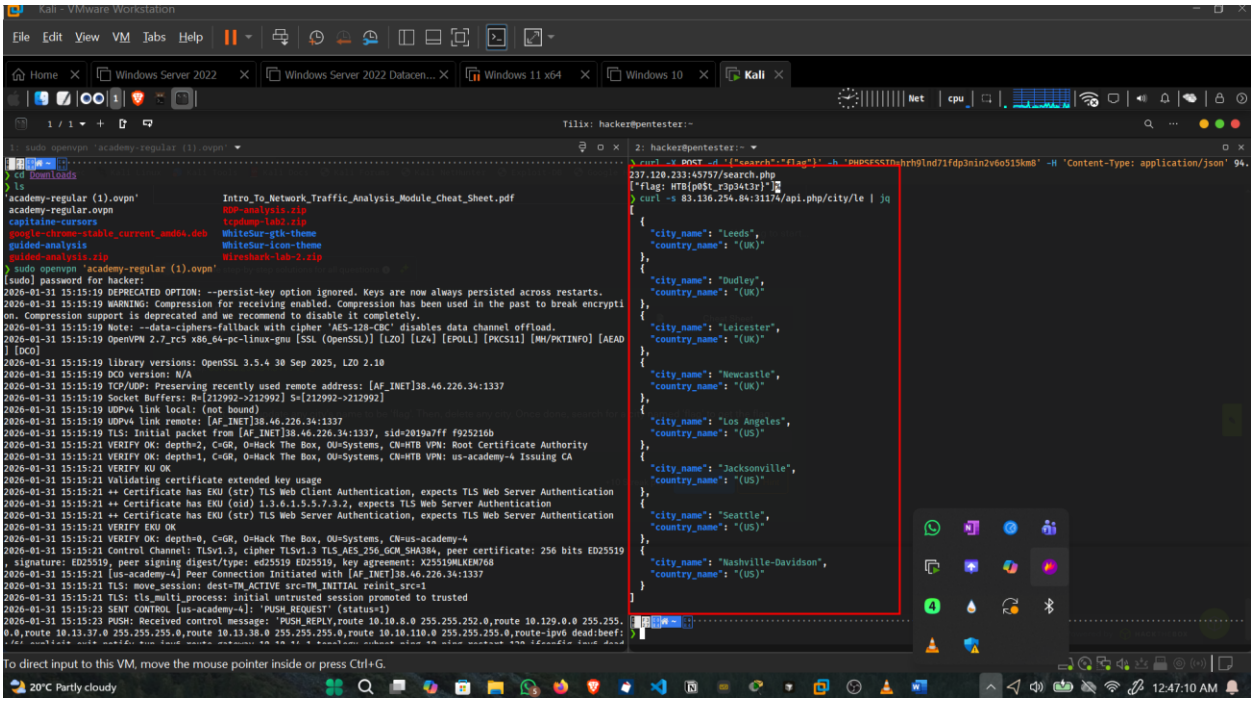
applications handle user input securely. Tools like cURL and browser developer tools were used to monitor and send POST requests.

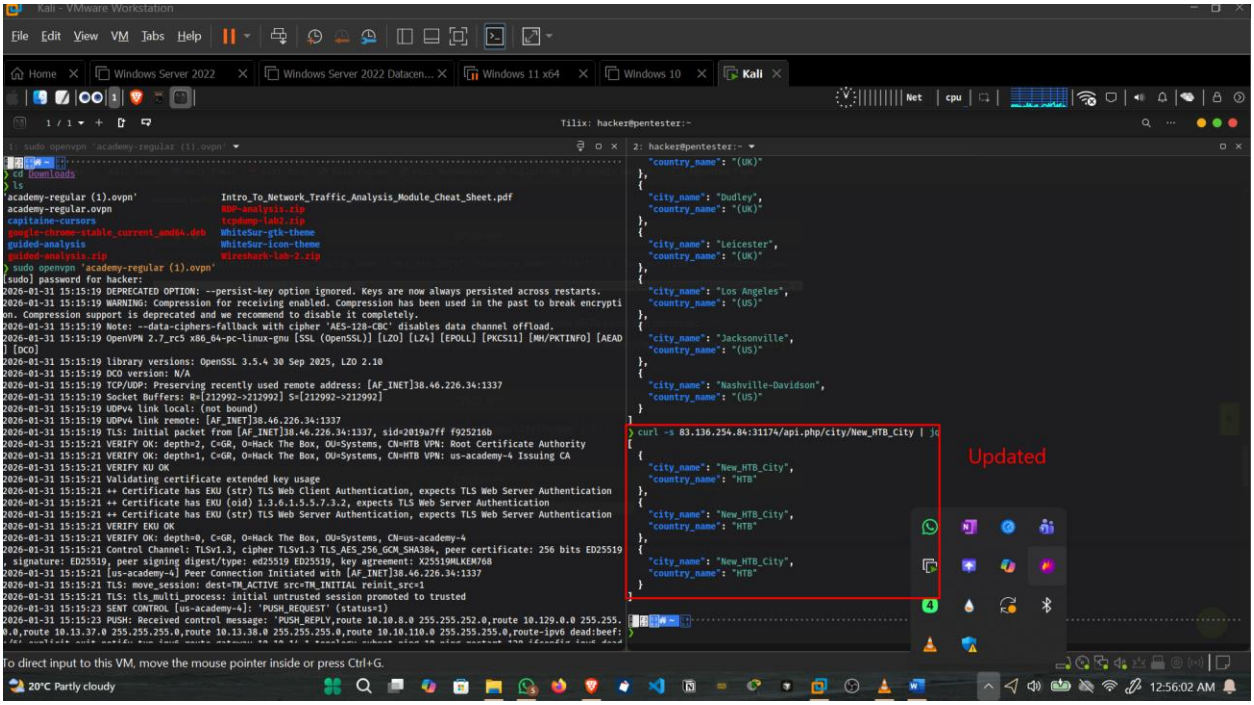


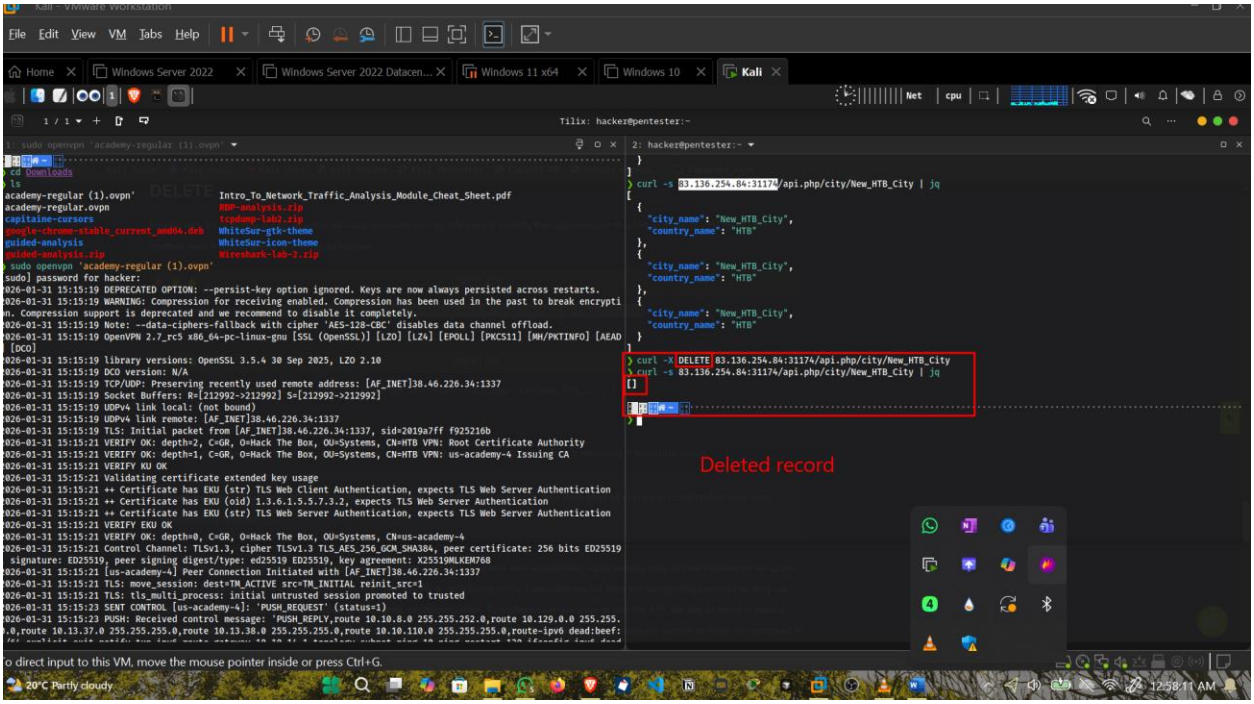


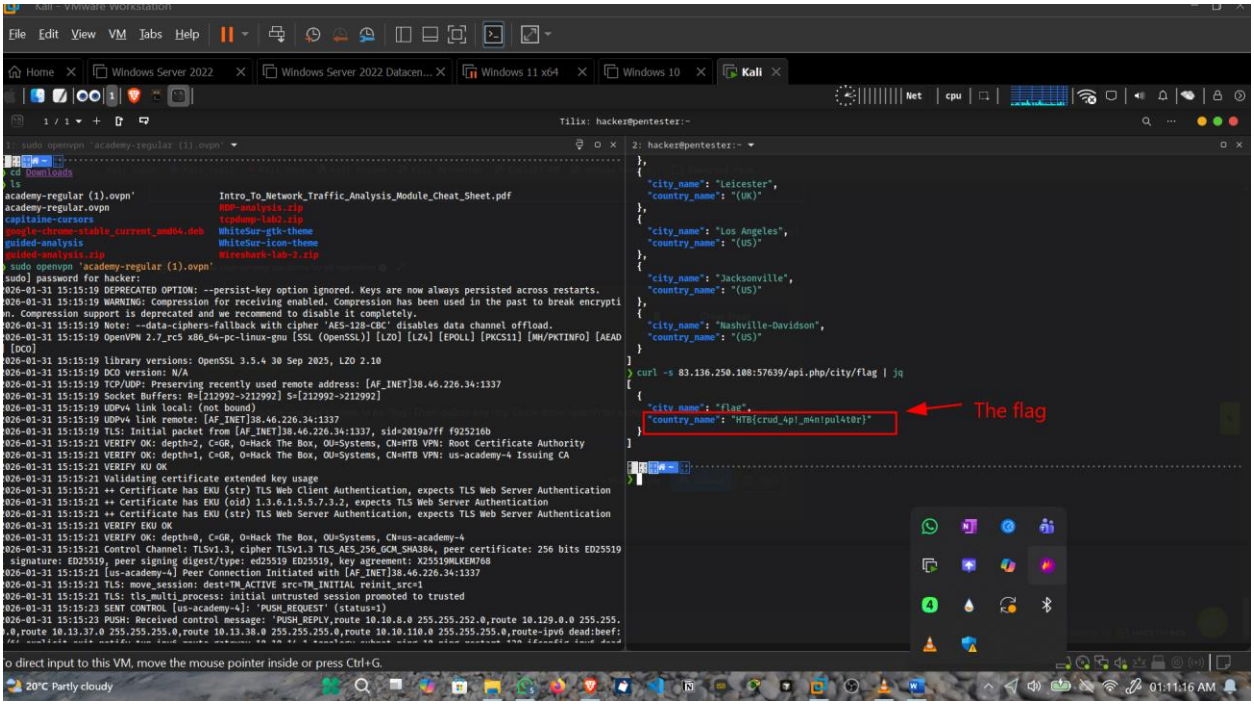
4.6 CRUD API

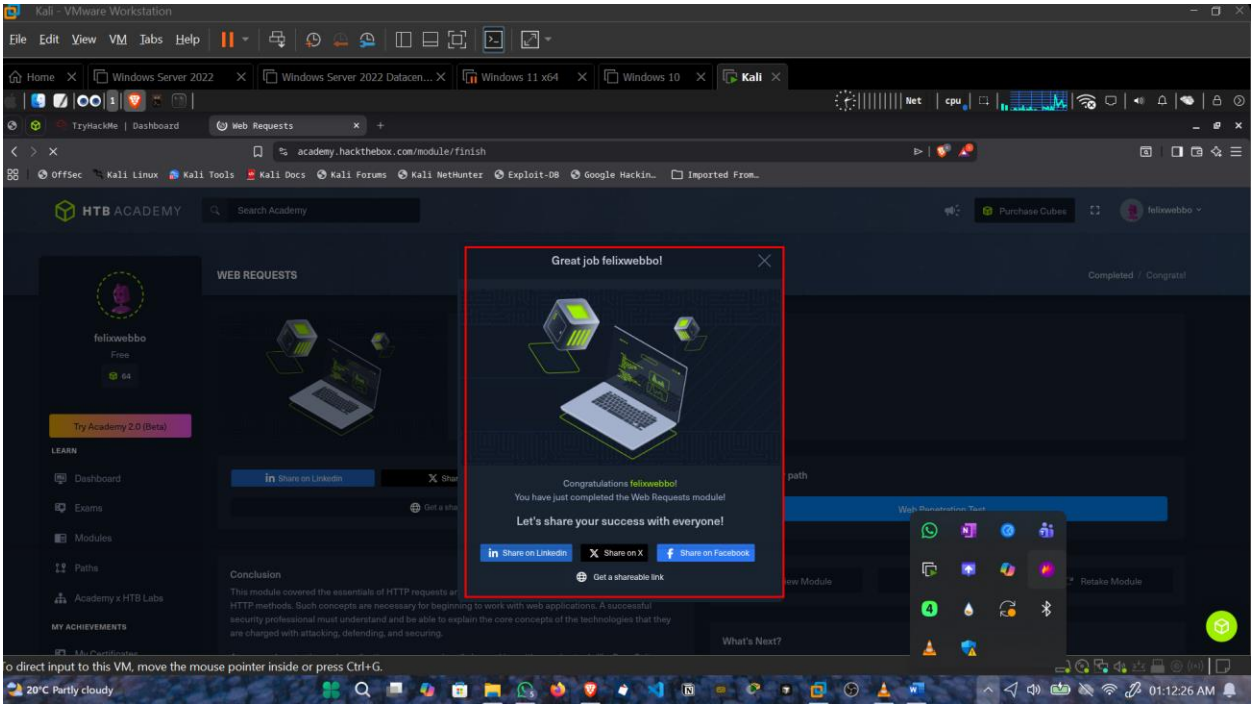
The module demonstrated how web applications use APIs to interact with databases. APIs allow clients to perform CRUD (Create, Read, Update, Delete) operations on specific resources using HTTP methods. For example, the api.php endpoint could be queried to update a city record in a database, specifying the table and row (e.g., updating the city “London”). This section highlighted direct API interaction and reinforced understanding of how backend operations are performed programmatically.




Updated


Deleted record





Visit here

CONCLUSION

The completion of the *Web Requests* module significantly enhanced understanding of HTTP and HTTPS communication, request structures, and web interaction mechanisms. Practical exposure

to cURL and browser Developer Tools improved confidence in analyzing and manipulating web traffic. The learning experience reinforced the importance of secure web communication and provided essential skills applicable to web application testing and penetration testing. Overall, the module served as a strong foundation for advanced web security analysis.

4.7 REFERENCES

1. Hack The Box Academy. (n.d.). *Web Requests Module*.
2. Mozilla Developer Network (MDN). *HTTP Overview*.