2025

Redynox

Intern: Uyor Chimdi Ebulu

Email: Chimdi2700@gmail.com

Objective

To develop hands-on understanding of web application security by analyzing common vulnerabilities using OWASP ZAP, performing basic vulnerability scans, and attempting manual exploitation of flaws such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF). This task provides a practical foundation in secure application assessment and ethical hacking techniques.

Skills: Basic Web Security, Vulnerability Identification

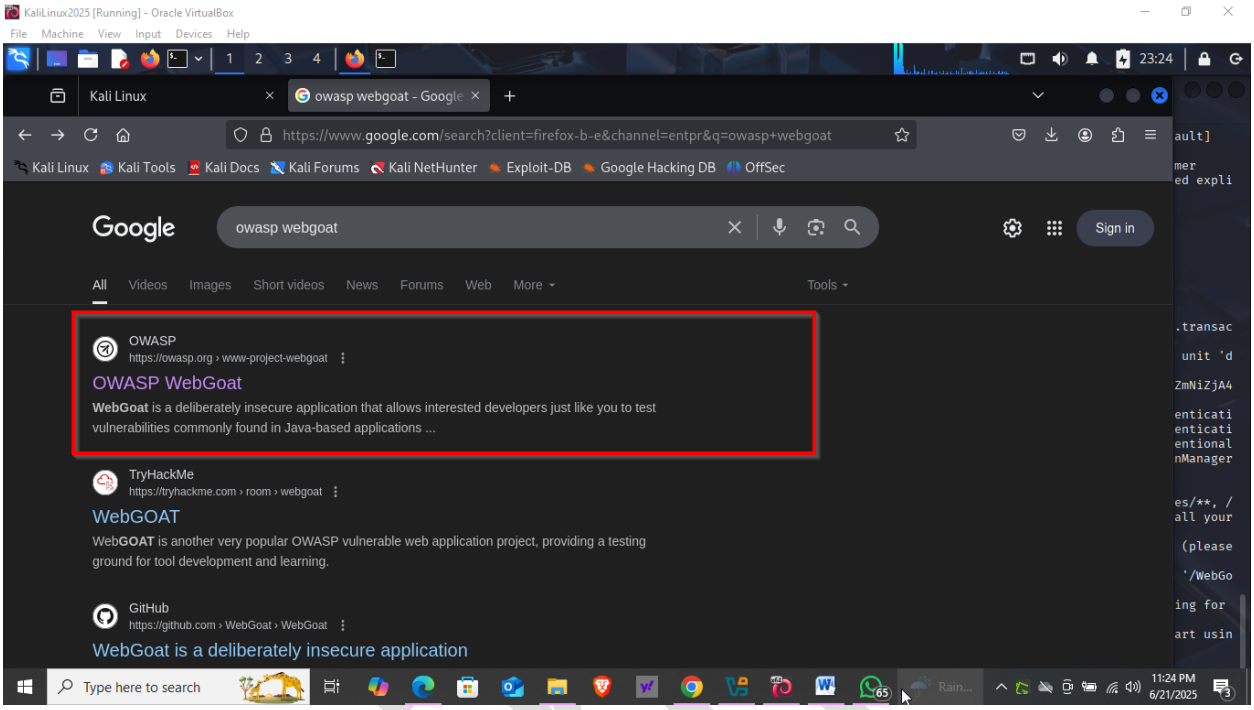
Tools: OWASP ZAP, WebGoat (or another vulnerable web application)

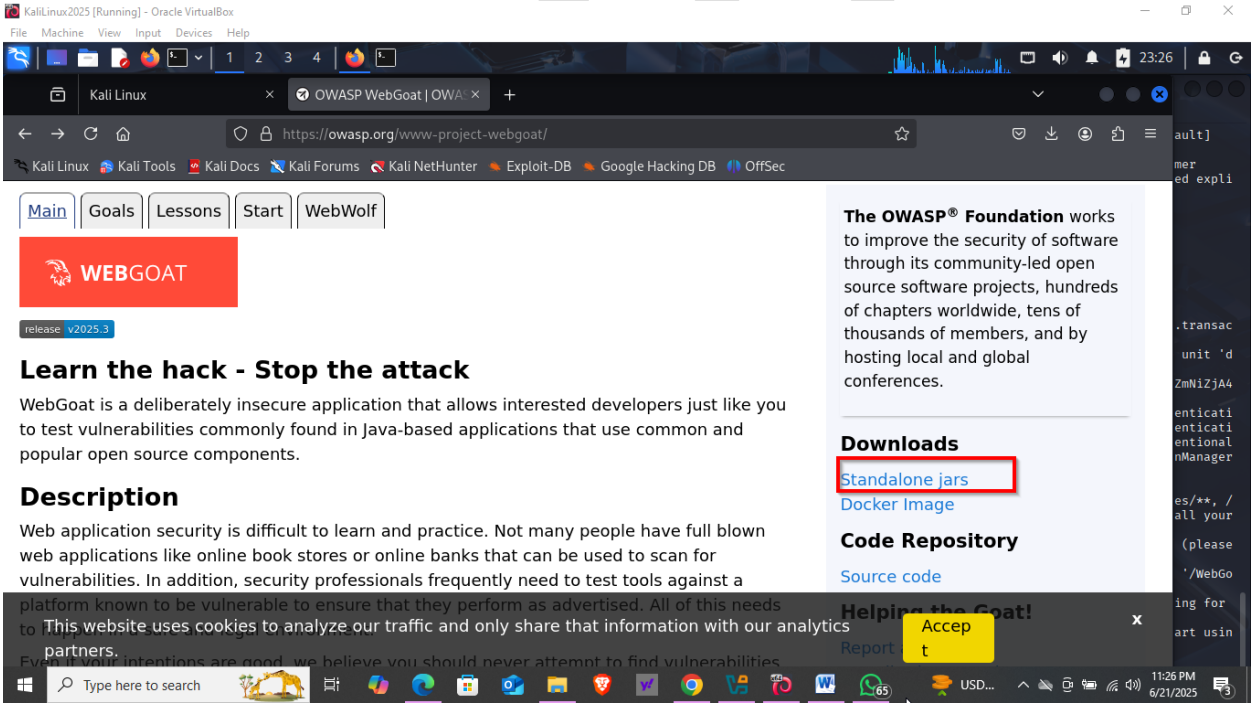
[CYBER SECURITY INTERNSHIP]

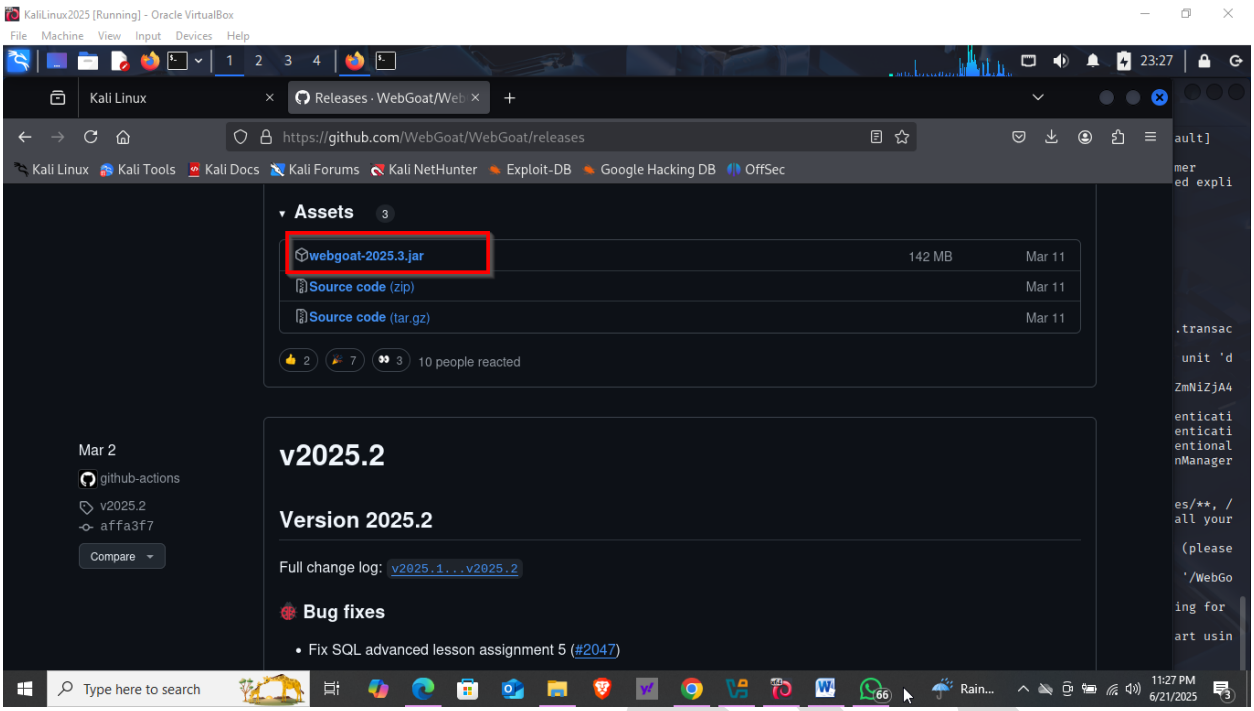
TASK 2: Introduction to Web Application Security

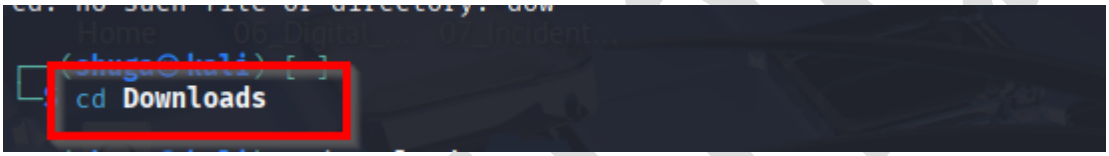
1. **Setup: 7/7/2025**

Installation of WebGoat

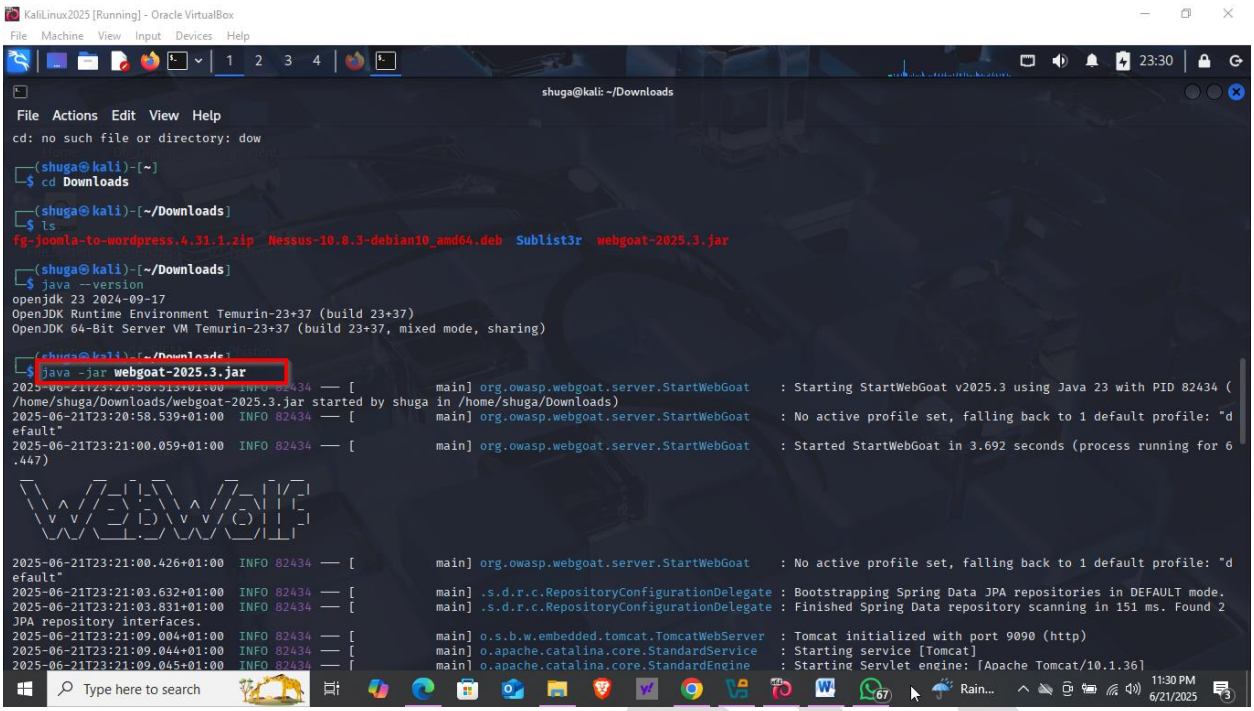




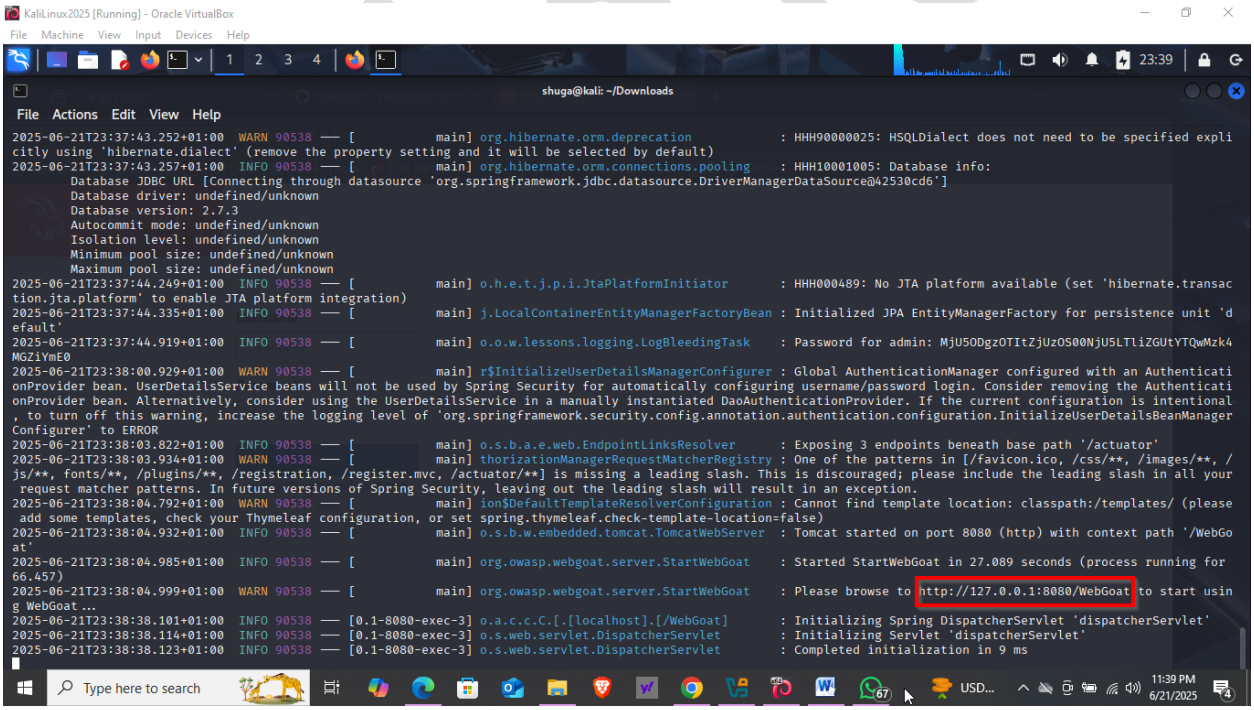




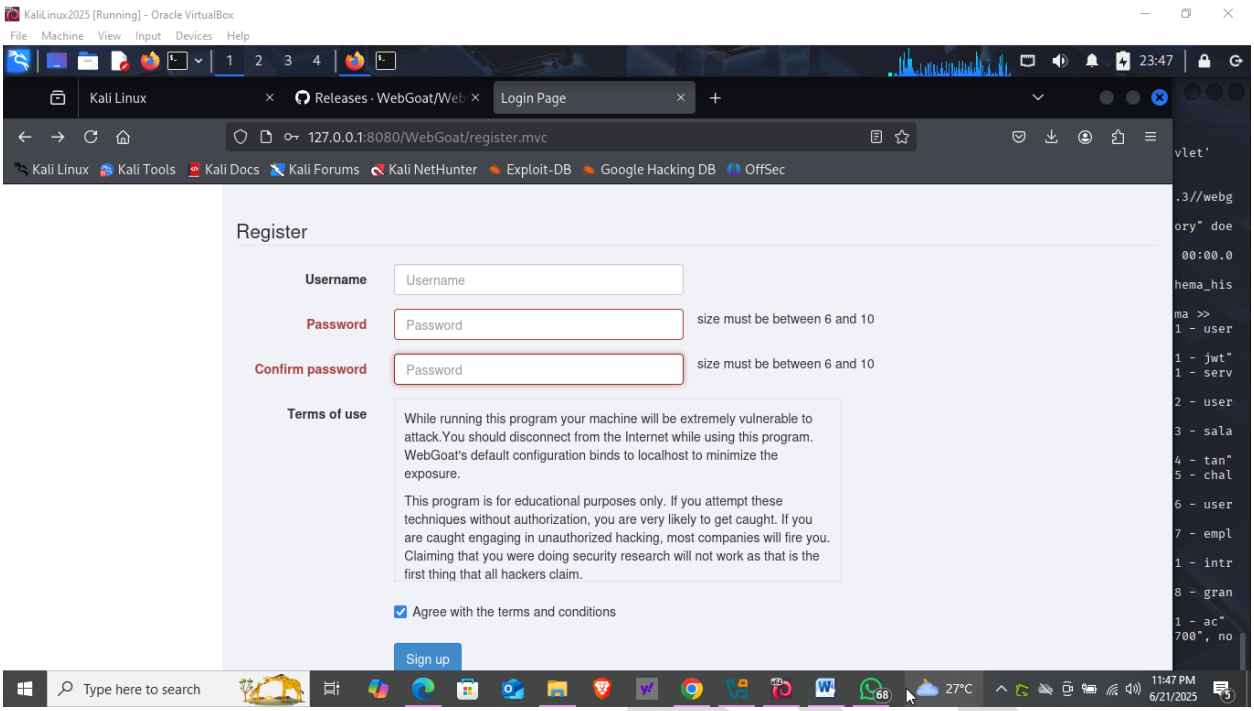
Checking the version of my java if it is up to date



Starting up the webgoat using java –jar webgoat-2025.3.jar



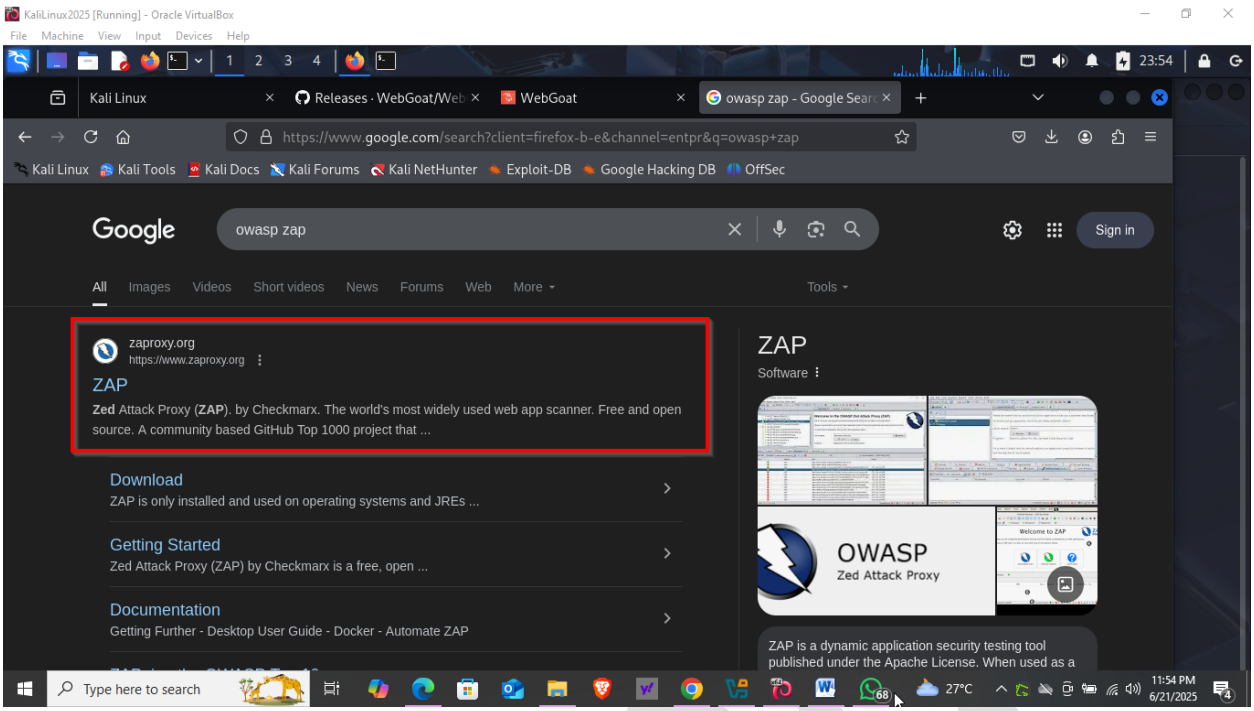
Launching the local host in the browser

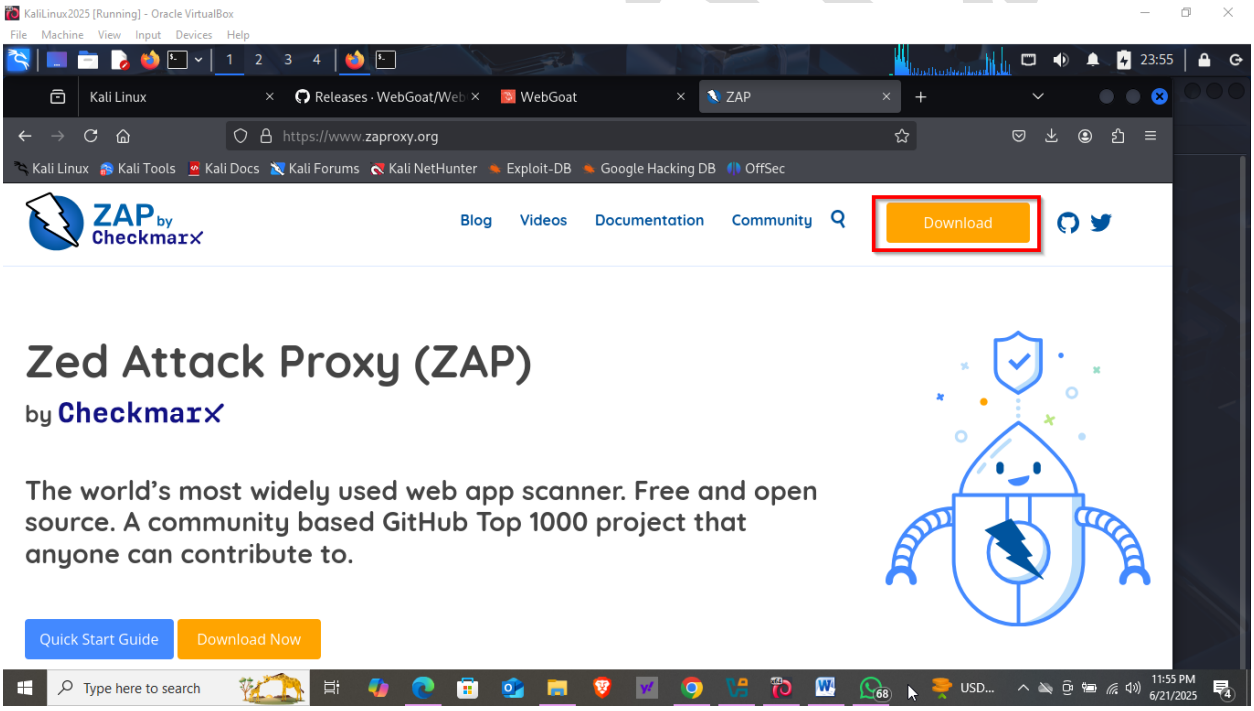


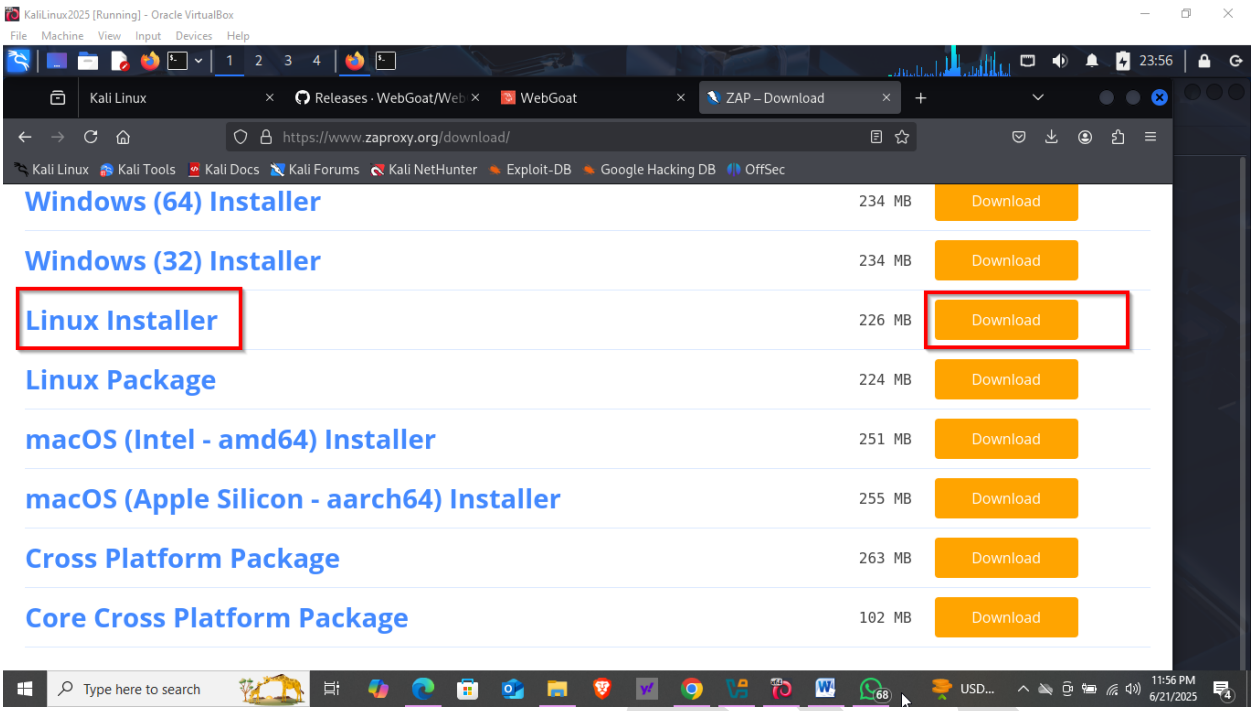
The signup page

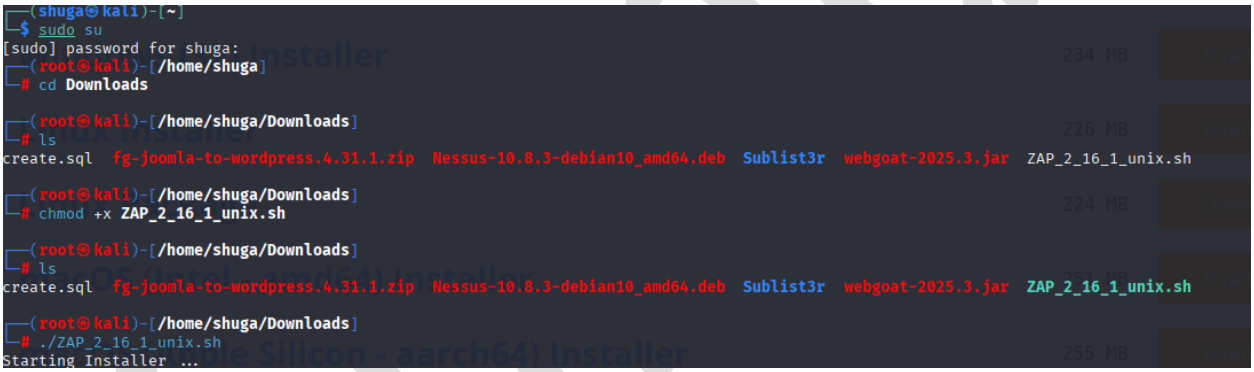
2. Perform Basic Vulnerability Analysis:

Installation of owasp zap

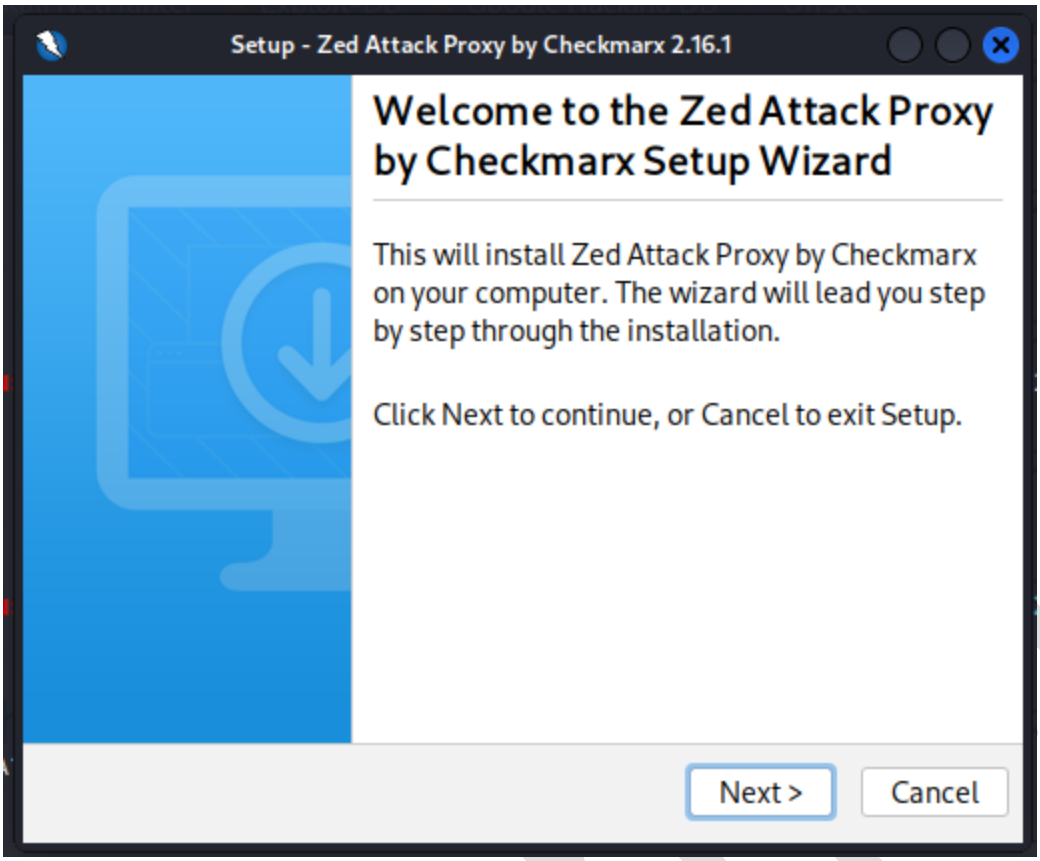

KaliLinux2025 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Kali Linux
Releases · WebGoat/Web
WebGoat
owasp zap - Google Sear
https://www.google.com/search?client=firefox-b-e&channel=entpr&q=owasp+zap
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
Google
owasp zap
Sign in
All Images Videos Short videos News Forums Web More
Tools
zaproxy.org
https://www.zaproxy.org
ZAP
Zed Attack Proxy (ZAP), by Checkmarx. The world's most widely used web app scanner. Free and open source. A community based GitHub Top 1000 project that ...
Download
ZAP is only installed and used on operating systems and JREs ...
Getting Started
Zed Attack Proxy (ZAP) by Checkmarx is a free, open ...
Documentation
Getting Further - Desktop User Guide - Docker - Automate ZAP
ZAP
Software
OWASP
Zed Attack Proxy
ZAP is a dynamic application security testing tool published under the Apache License. When used as a
Type here to search
27°C
11:54 PM
6/21/2025

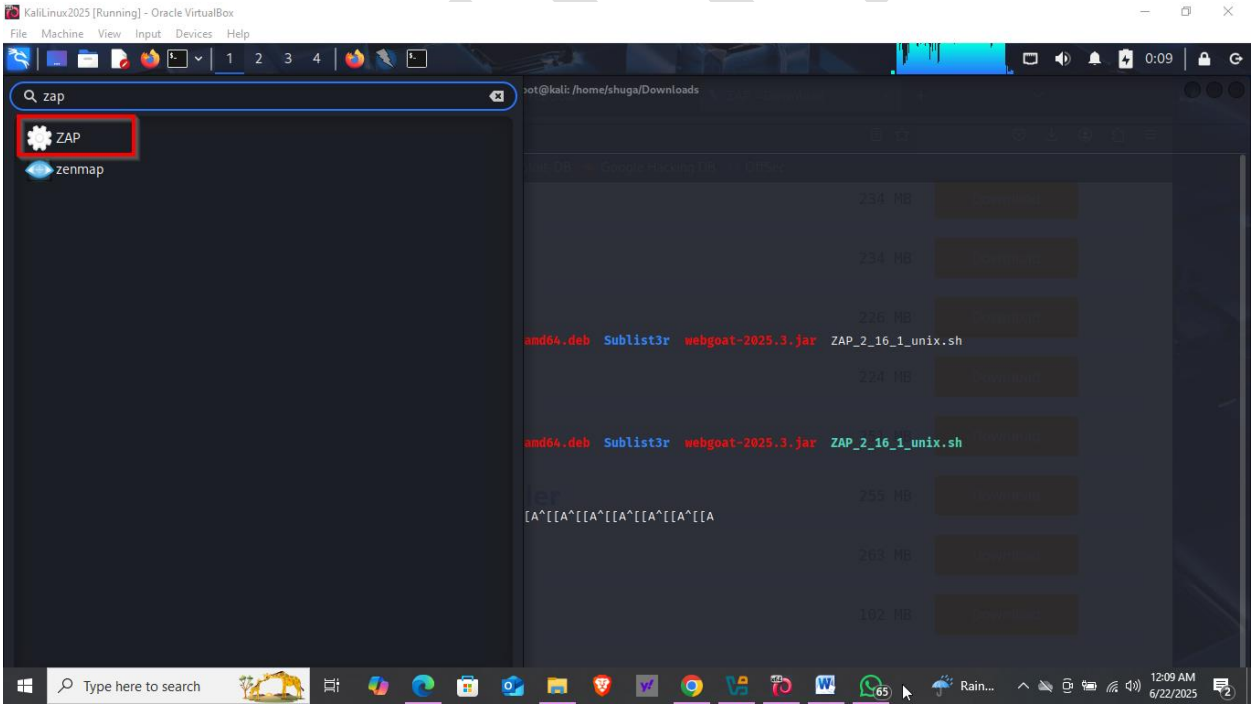

KaliLinux2025 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Kali Linux
Releases · WebGoat/Web
WebGoat
ZAP
https://www.zaproxy.org
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
ZAP by Checkmarx
Blog Videos Documentation Community
Download
Zed Attack Proxy (ZAP)
by Checkmarx
The world's most widely used web app scanner. Free and open source. A community based GitHub Top 1000 project that anyone can contribute to.
Quick Start Guide
Download Now
Type here to search
USD...
11:55 PM
6/21/2025



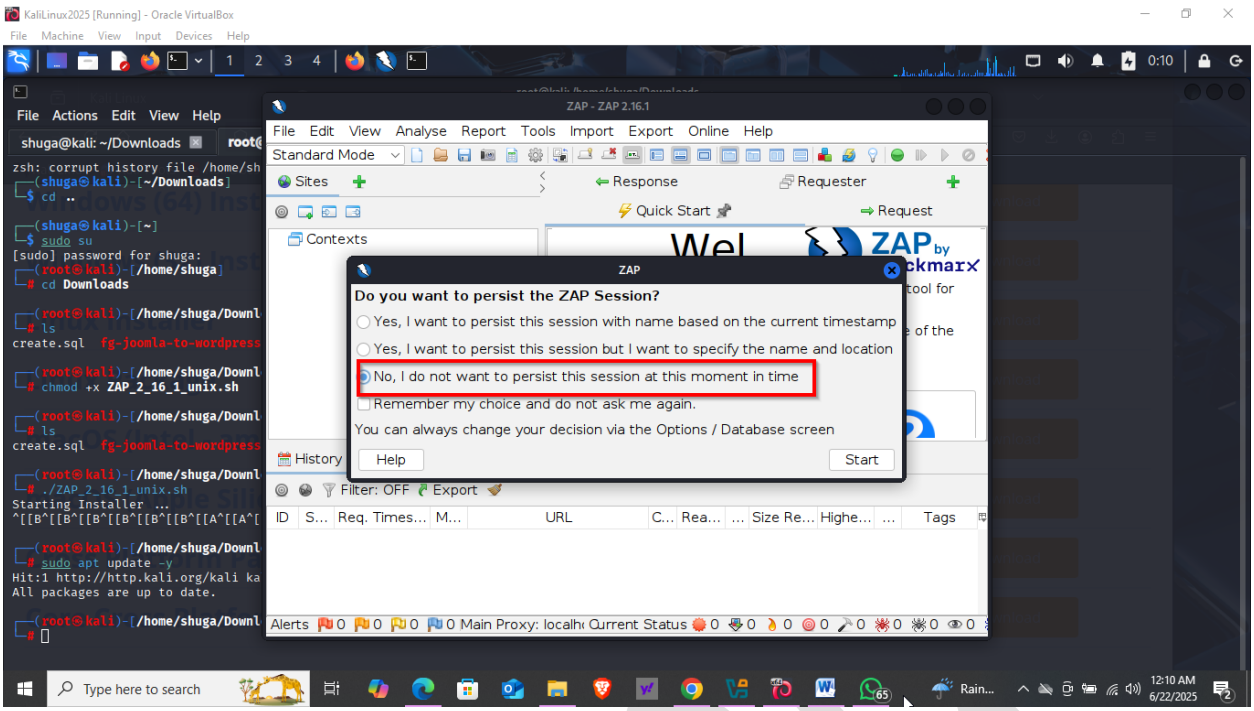


- I switch to the super user (root) account using sudo su to allows a permitted user to execute a command as the super user or another user .
- I changes the directory to `Downloads` within `my` home directory.
- I use the `ls` command to list the contents of the current directory.
- I uses the chmod command to change the permissions of the file ZAP_2_16_1_unix.sh.
- `+x` adds execute permission to the file for the owner, group, and others. This is a common step before running a shell script.
- I use ./ZAP_2_16_1_unix.sh to start the installer of OWASP ZAP.

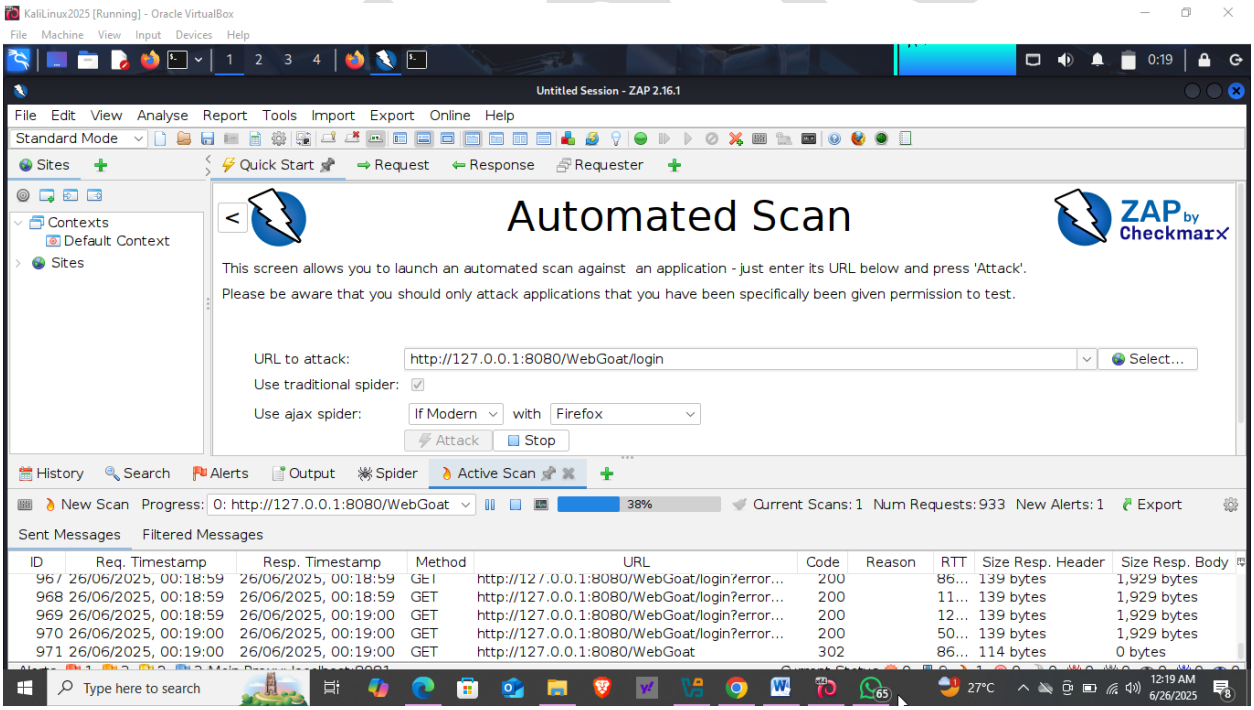




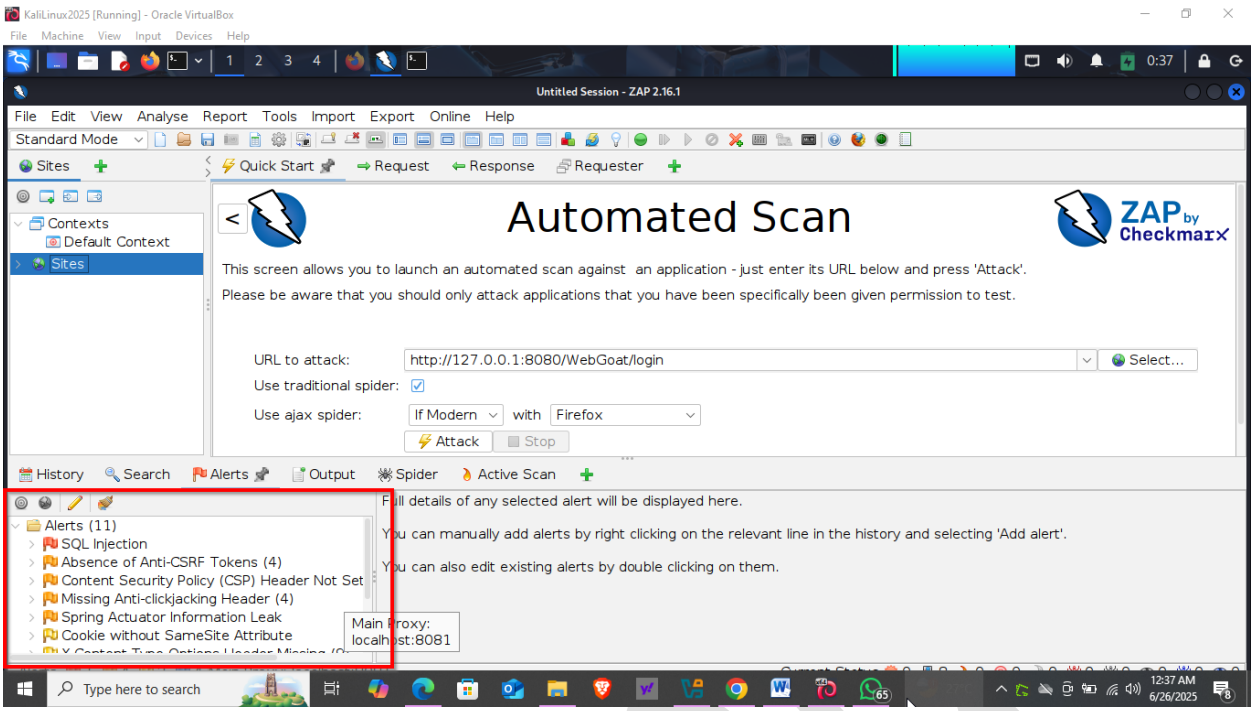
Searching for ZAP and Launching ZAP



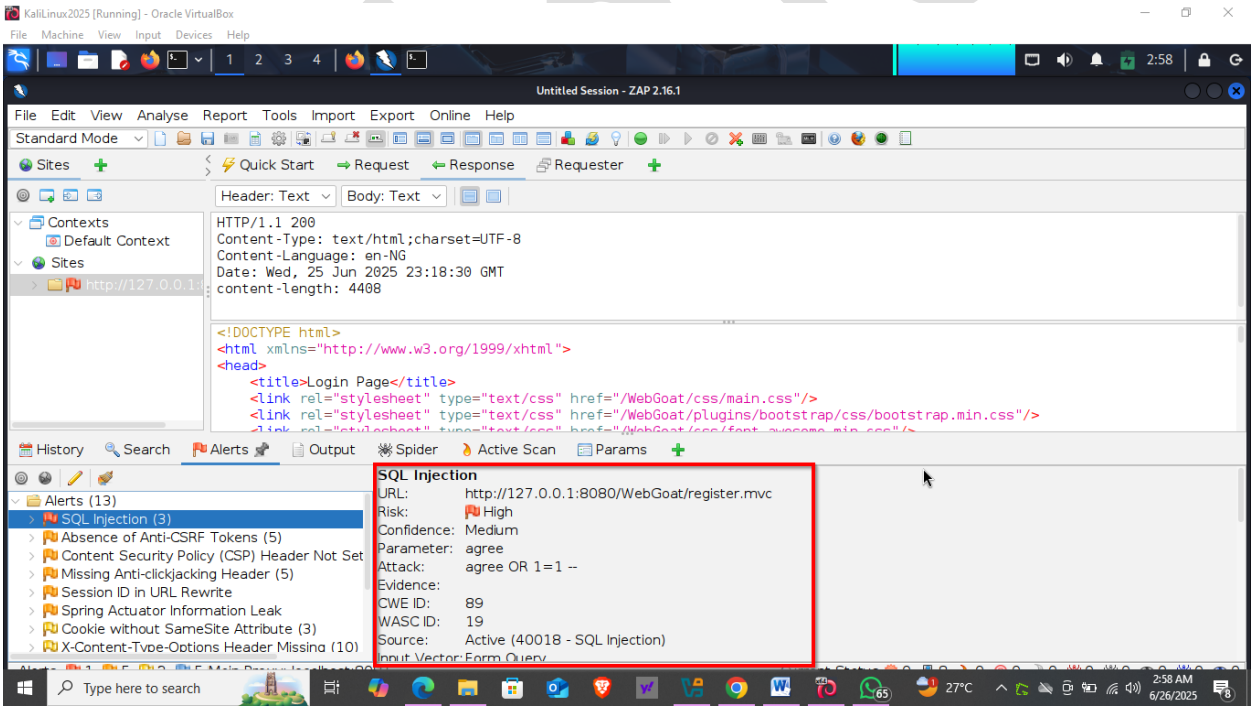
Session Persistence



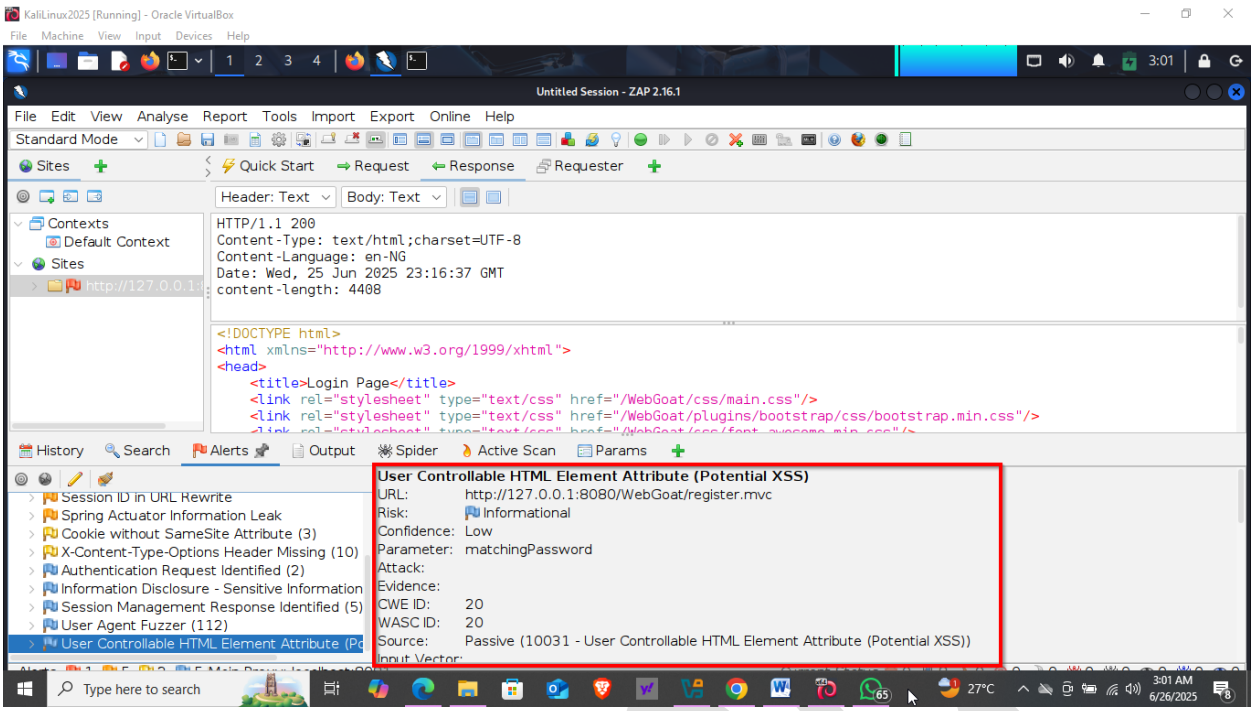
Performing a Scan



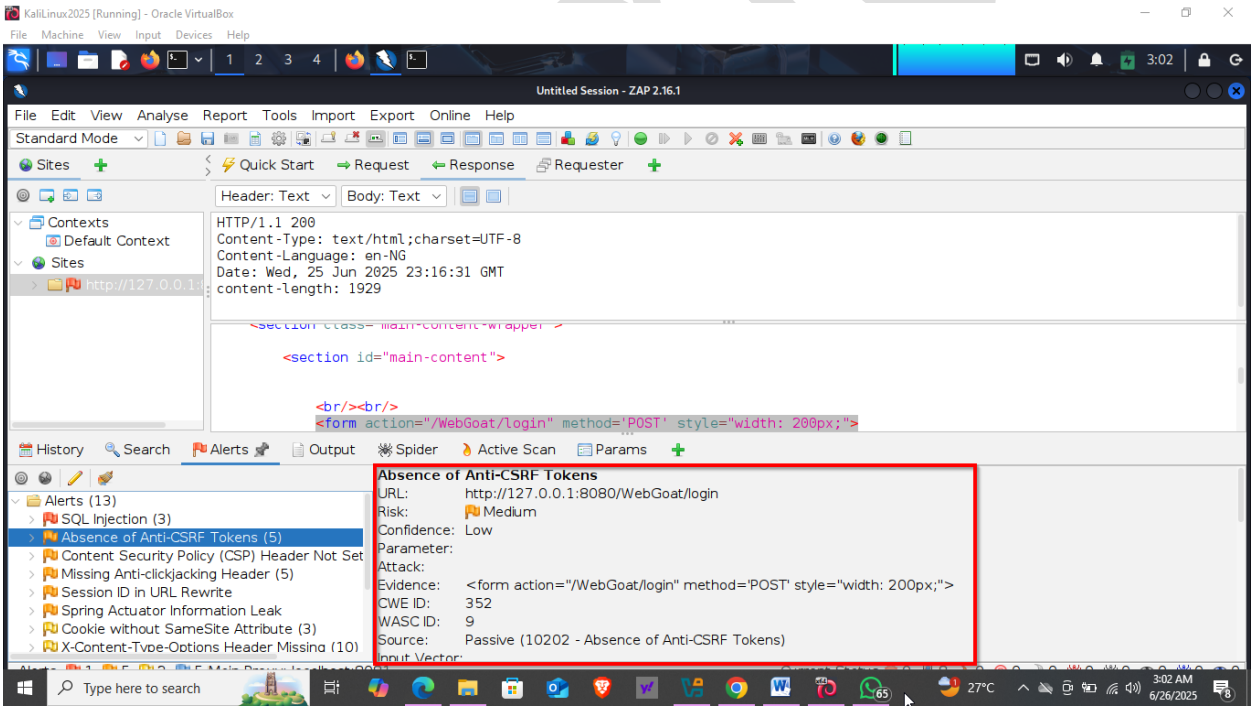
The vulnerability results



The SQL injection is high



This is the Cross-Site Scripting (XSS)



This is Cross-Site Request Forgery (CSRF)

3. Explore Vulnerabilities:

- **Understand how each vulnerability works by reading the descriptions provided by OWASP ZAP.**

1. SQL Injection

Description from OWASP ZAP:

SQL Injection occurs when user-supplied input is inserted into a SQL query without proper validation or sanitization. This allows attackers to manipulate the query to:

- Bypass authentication
- Retrieve, modify, or delete database data
- Execute administrative operations on the database

2. Cross-Site Scripting (XSS)

Description from OWASP ZAP:

XSS flaws occur whenever an application includes untrusted data in a web page without proper validation or escaping. Attackers can execute malicious scripts in the user's browser, enabling:

- Session hijacking
- Website defacement
- Redirection to malicious sites

Types:

- **Reflected XSS**: Payload is reflected in the response (e.g., in URLs or search results).
- **Stored XSS**: Payload is permanently stored (e.g., in comments or database) and shown to users.
- **DOM-based XSS**: Client-side JavaScript mishandles user input and modifies the DOM.

3. Cross-Site Request Forgery (CSRF)

Description from OWASP ZAP:

CSRF occurs when a user is tricked into performing an action in a web app without their consent. It abuses the user's active session with the target application.

Example:

A logged-in user visits a malicious page that silently sends a request to change their email or password on another site, using their session cookies.

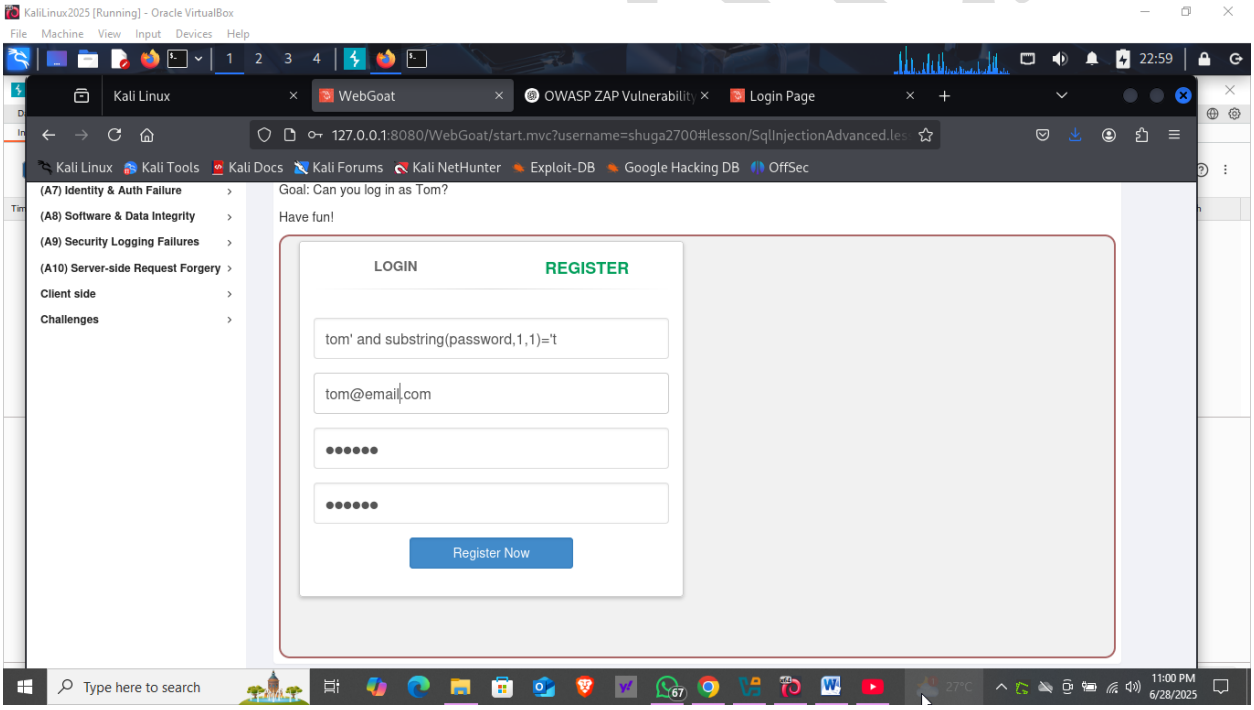
Missing Protection:

Applications are vulnerable if they do not use anti-CSRF tokens (unique, unpredictable values in form submissions or headers) to validate requests.

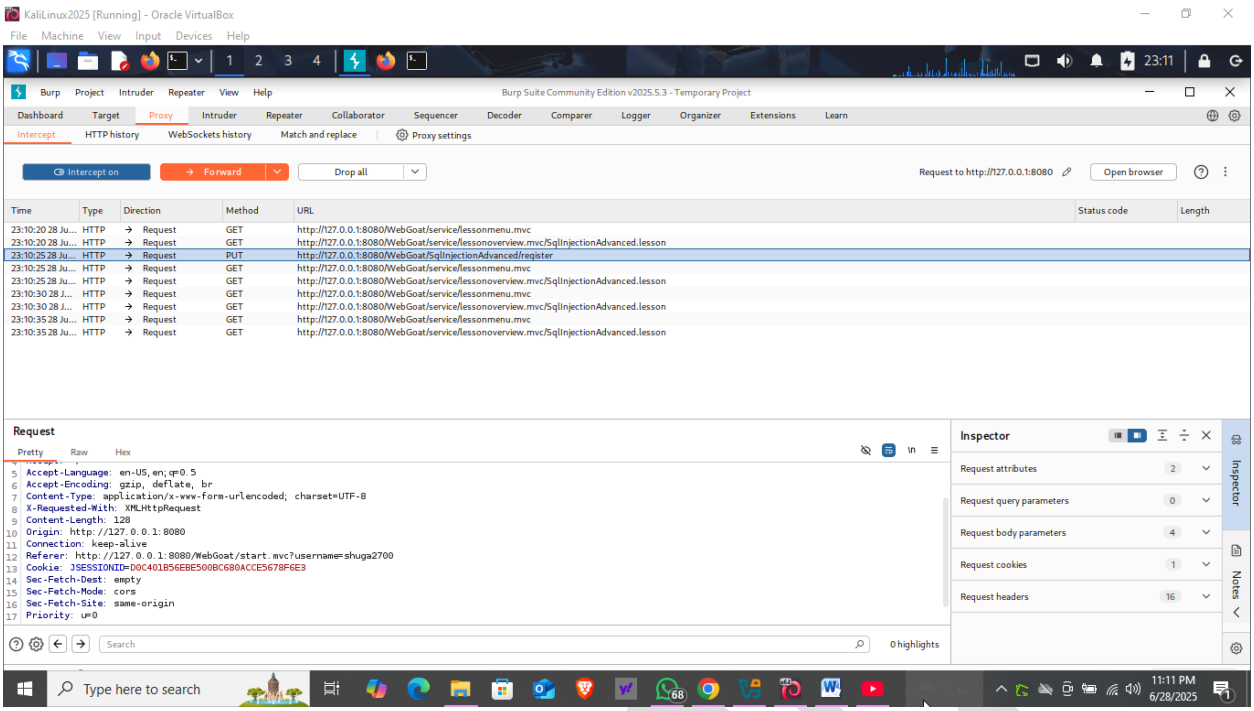
Summary of ZAP Alert Descriptions

Vulnerability	ZAP Description Summary	Common Consequences
SQL Injection	Unvalidated input used in SQL queries	Auth bypass, data leaks, DB manipulation
XSS	Injected scripts run in user's browser	Session theft, redirection, defacement
CSRF	Malicious actions via user's session	Unauthorized account changes, data modification

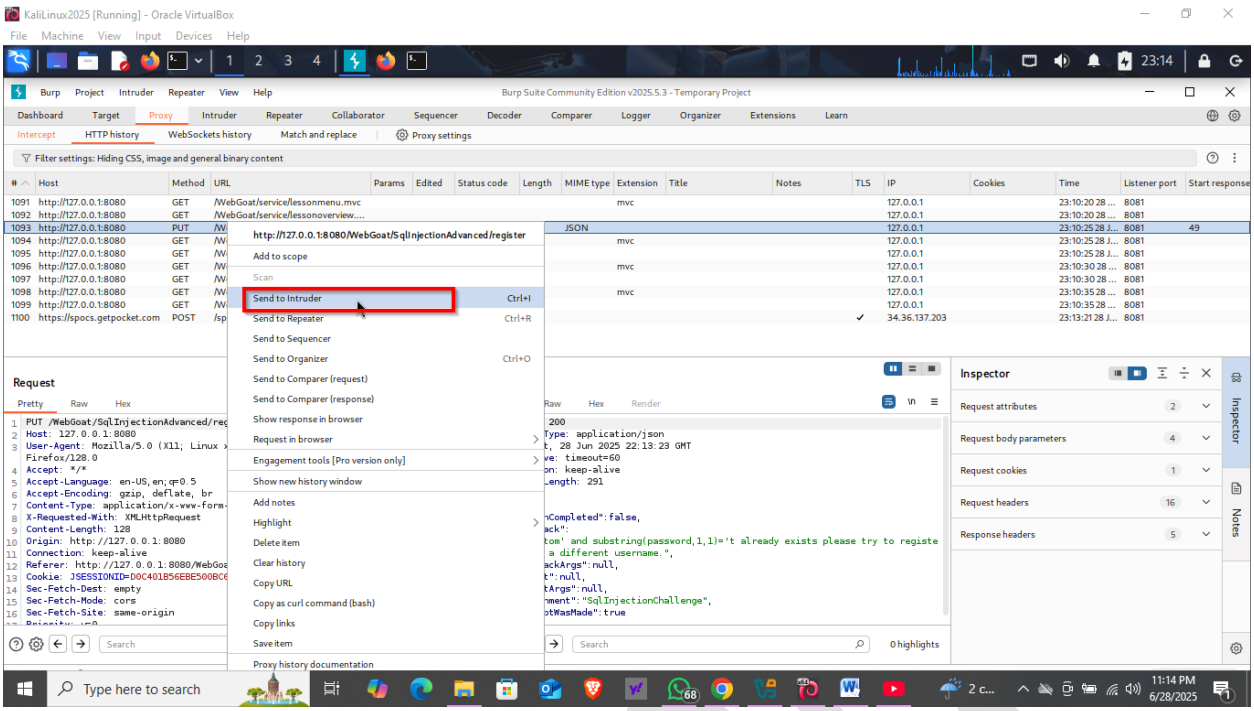
- Attempt to manually exploit these vulnerabilities using basic techniques (e.g., inserting SQL code in a login form).



The register user name field is vulnerable to an SQL injection attack and I got a reply saying user name already exists message.

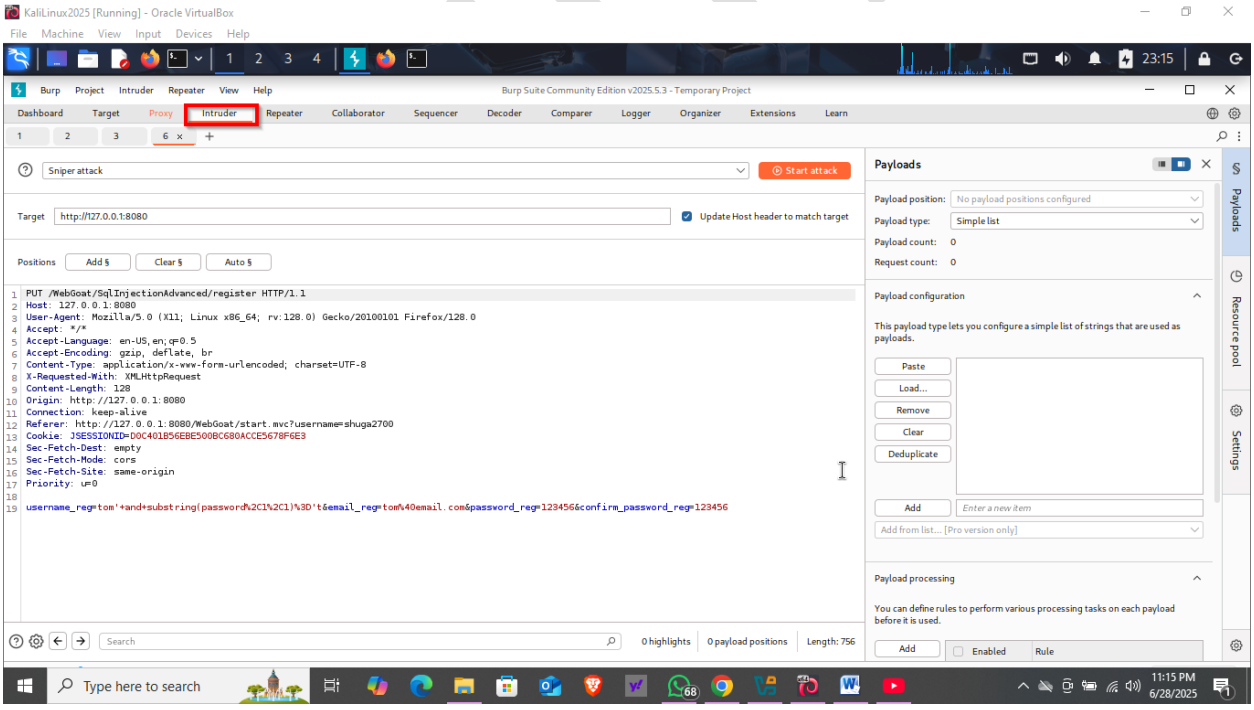


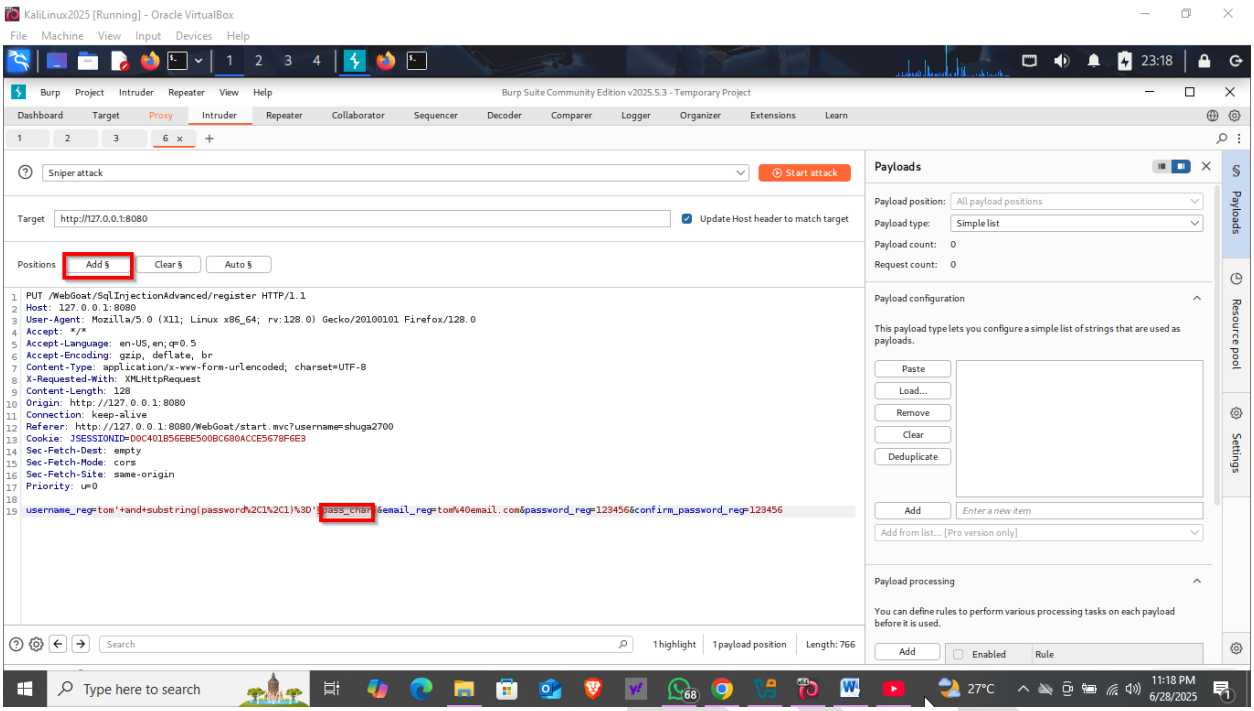
I used Burp Suite for convience and find that the password is 23 characters long. Now, we need to find the password itself. We can achieve that by using SQL injection to test each character of the password individually. `tom' AND substring(password, 1, 1) = 't`. I click on the Register now button in the webgoat which the input in the burp suite Method is PUT.



Go to Burp proxy HTTP history

Right click on PUT – sent to intruder

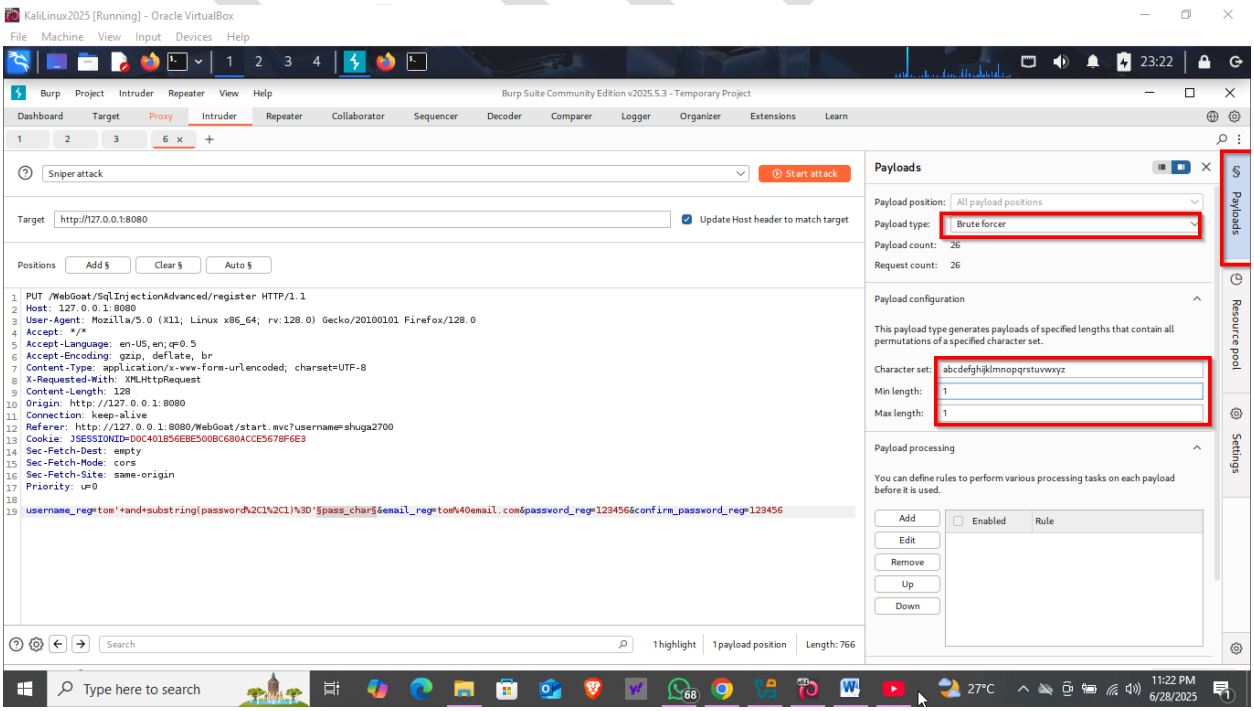




Click on intruder – positions then clear button

Replace ‘t’ with pass_char

Highlight pass_char and click Add button



Click payloads

Now, we will arrange the payloads for the Burp Suite cluster bomb attack. The first set of payloads will test the character positions from 1 to 26 to identify the correct position of each character.

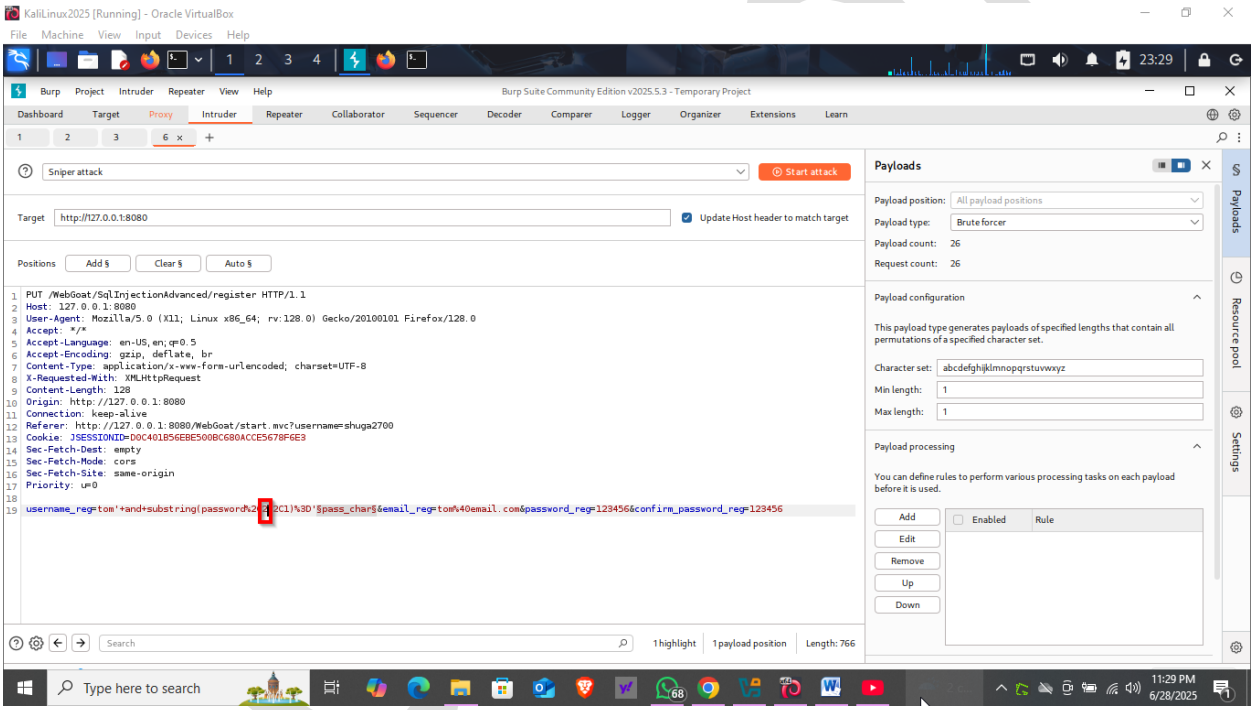
Setup as shown

Click ‘start attack’

When finished click ‘Length’ twice

The first letter of the password is ‘t’.

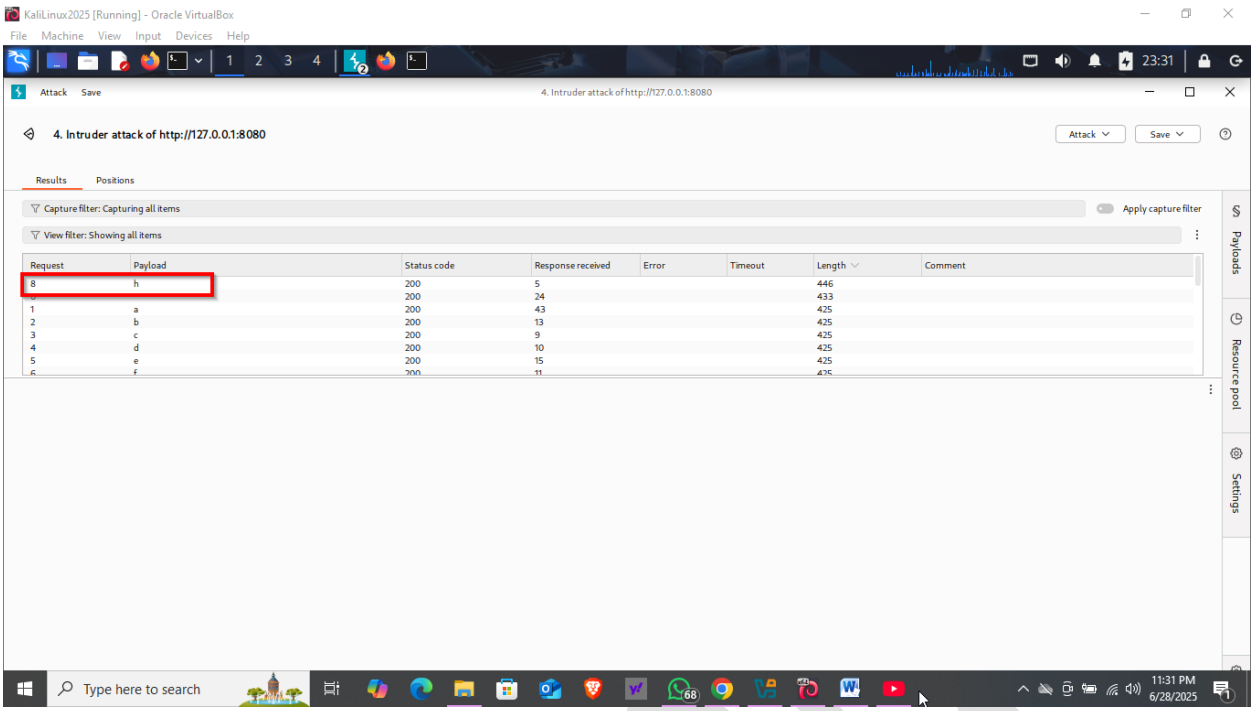
Close Attack window



Go back to positions

Change ‘1’ to ‘2’

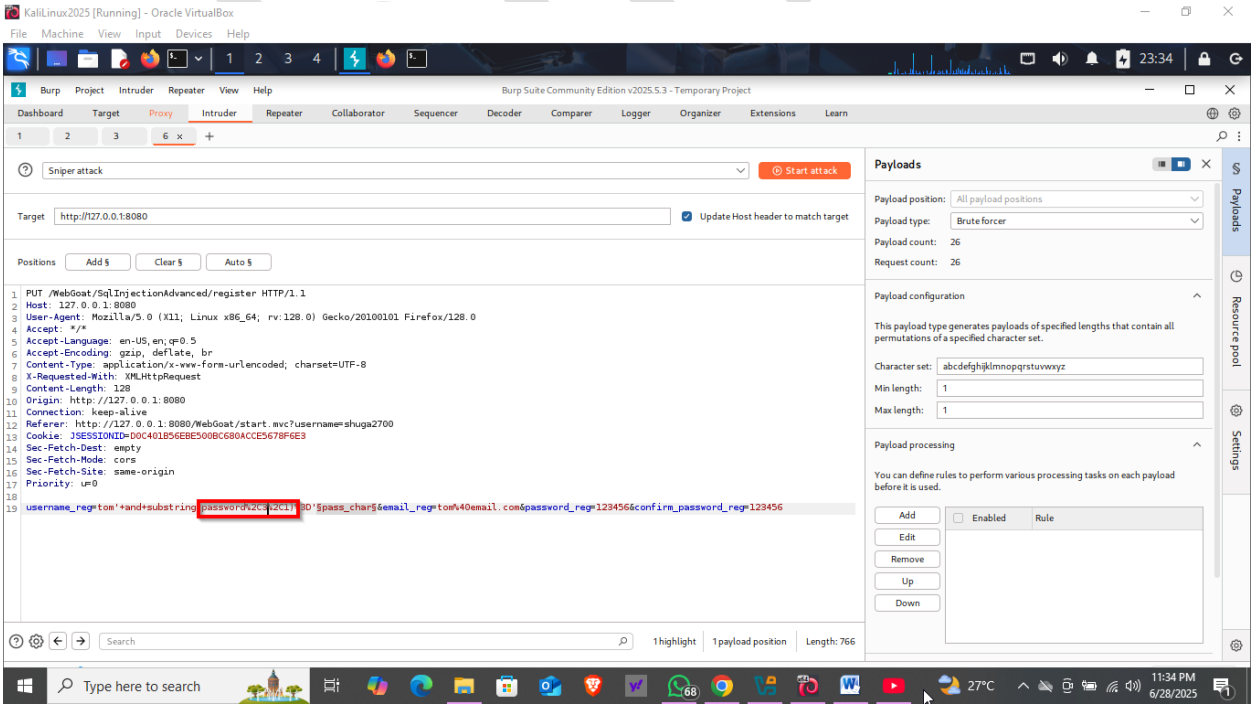
Click ‘start attack’



When finished click ‘Length’ twice

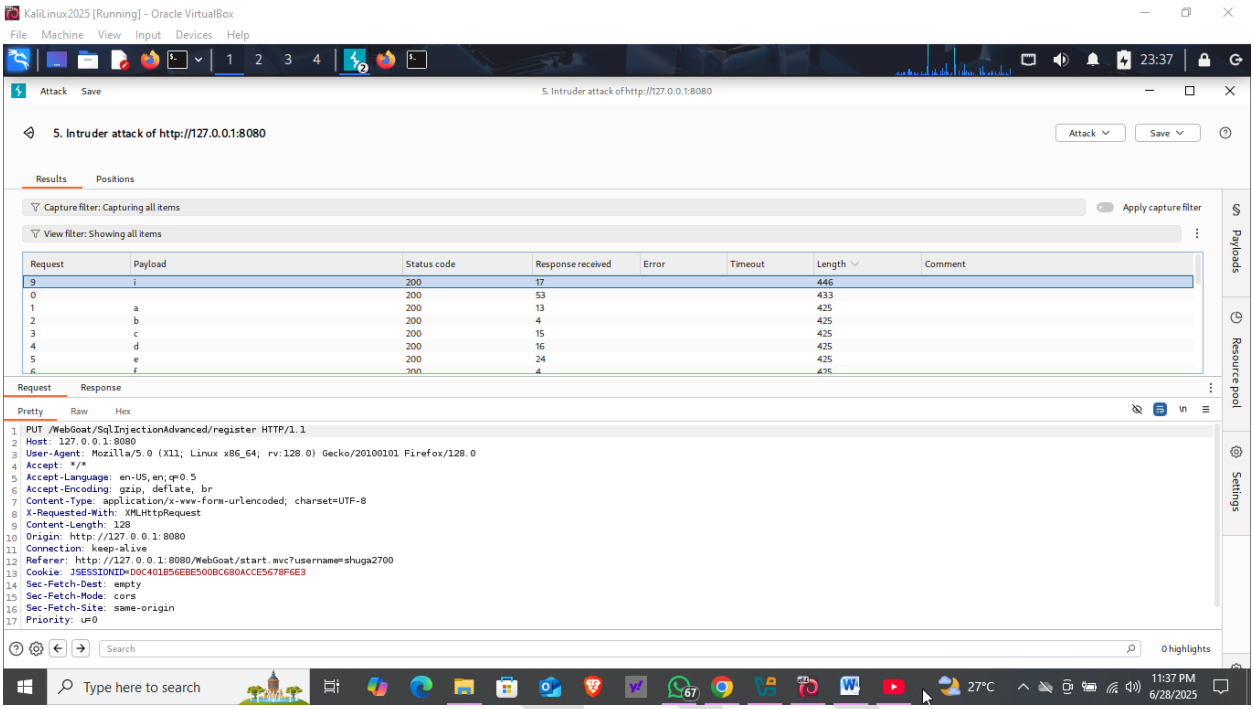
The second letter of the password is ‘h’

Close Attack window



In positions change ‘2’ to ‘3’

Click ‘start attack’

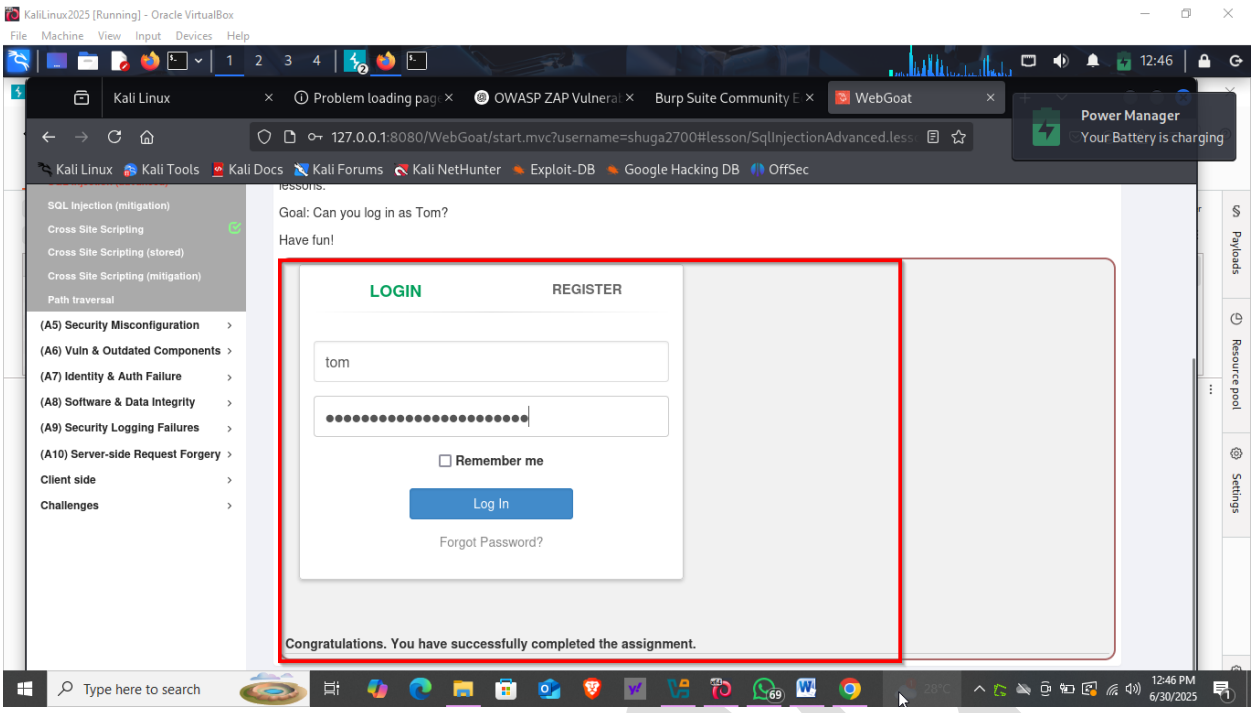


When finished click ‘Length’ twice

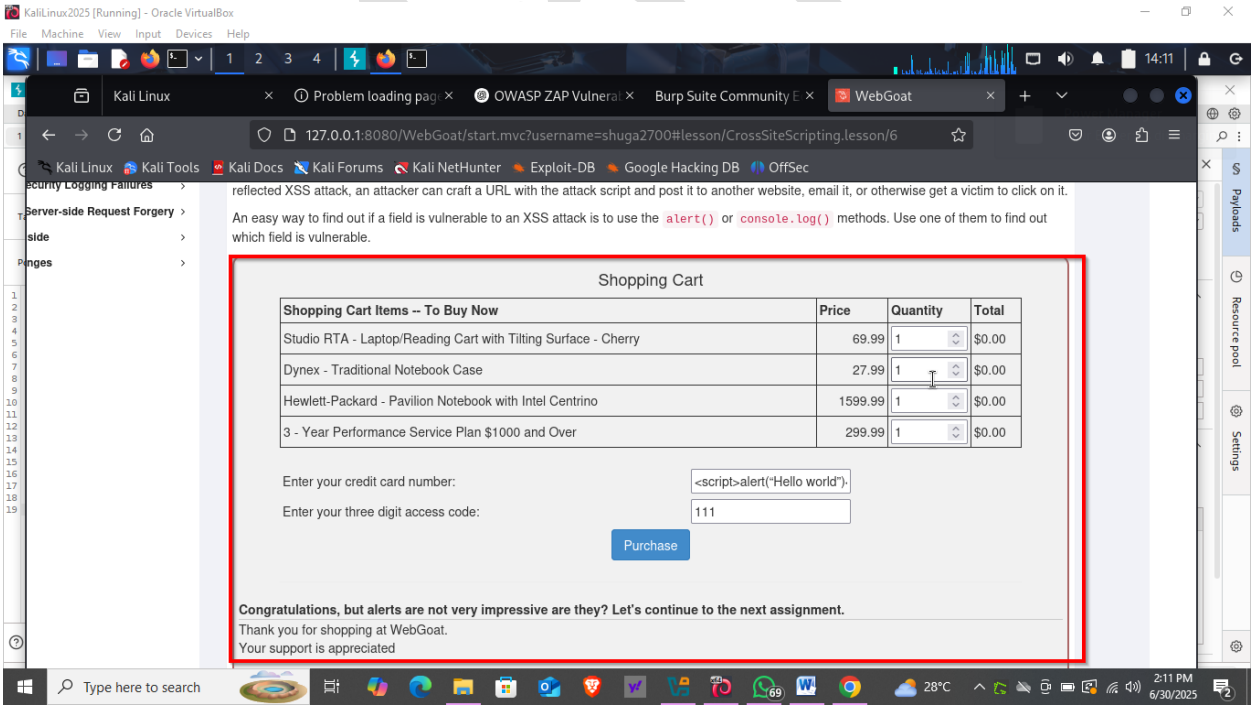
Third letter of password is ‘I’

Close Attack window

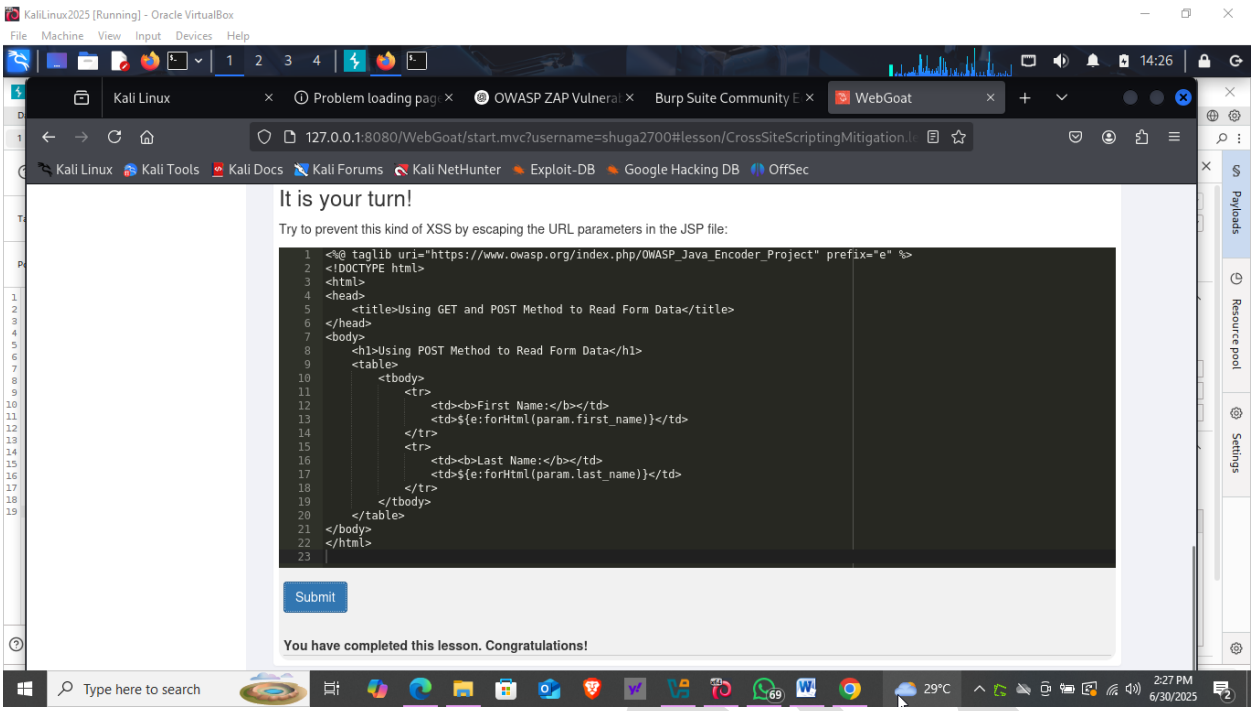
Repeat until you have the full password.



After the attack is complete, we obtain the password `thisisasecretfortomonly`



Let's start by creating a simple XSS payload to test for vulnerabilities in the application. In this case, we use the payload `<script>alert("Hello world")</script>`, which is a basic script injection designed to trigger an alert box in the browser.



Reflection and Learning

This task emphasized the importance of understanding:

- **How vulnerabilities work technically**
- **How attackers exploit poor validation**
- **How to use tools like ZAP to automate and assist testing**
- **Why defense-in-depth (tokenization, sanitization, session control) is crucial**

By combining automated scans with manual testing, a cybersecurity analyst gains deeper insight into web application behavior and potential attack vectors.

Conclusion

Task 2 provided real-world insight into web application vulnerabilities and their exploitation using OWASP ZAP. It reinforced the value of continuous learning and the use of industry-standard tools for proactive vulnerability assessment. Understanding and mitigating such threats is key to ensuring application security in any organization.