NETWORK AND WEB APPLICATION SECURITY REPORT

Submitted by: Suresh Babu
Date: July 2025

Introduction

As cyber threats continue to evolve and challenge the security of modern digital systems, gaining practical exposure to real-world cybersecurity concepts is crucial for any aspiring cybersecurity professional. During my internship at Redynox, I had the opportunity to work on a series of hands-on tasks that introduced me to core areas of cybersecurity, including network security, web application security, and professional engagement within the cybersecurity community.

This report outlines my learning journey through three key assignments:

- **Task 1**: Understanding and applying network security fundamentals by identifying threats, configuring firewalls, and analyzing traffic using Wireshark.
- **Task 2**: Exploring web application vulnerabilities using OWASP ZAP and WebGoat to identify and exploit common security flaws such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).
- **Task 3**: Strengthening my professional presence on LinkedIn by engaging with cybersecurity-related content and promoting awareness about secure digital practices.

Through these activities, I developed essential technical skills and gained insights into how basic security measures can defend systems against common attacks. This internship provided me with a foundational understanding of vulnerability assessment, secure configurations, and the importance of security awareness — vital competencies for a successful career in cybersecurity.

Task 1: Introduction to Network Security Basics

1. Network Threats Summary

Virus: Malicious software that attaches itself to files and spreads when they are executed.
Worm: Self-replicating malware that spreads across networks without needing to attach toiles.
Trojan: Malicious software disguised as legitimate applications to trick users into installing it.
Phishing: Social engineering attack where users are tricked into providing sensitive information through fake websites oremails.

2. Security Measures Implemented

To protect my home network, I implemented the following basic security measures:

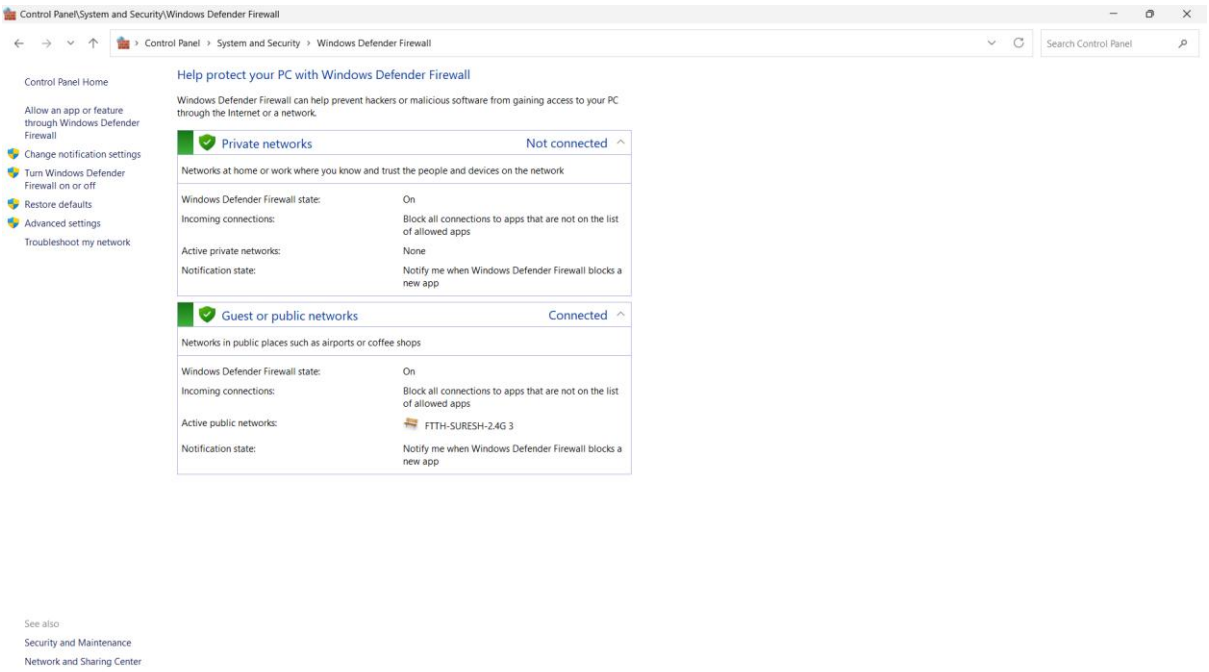
- Enabled Windows Defender Firewall and ensured it was active on all profiles (Domain, Private, Public).
- Created a custom rule to block an unused port (e.g., port 23 for Telnet).
- Accessed my home router settings, changed the default admin password to a strong one.
- Enabled WPA2 encryption for Wi-Fi to ensure secure wireless communication.
- Disabled WPS and UPnP for better protection against unauthorized access.

These changes help prevent common attacks such as brute-force login, man-in-the-middle, and open-port scanning.

1. Enable Windows Defender Firewall (All Profiles)

1. Press Windows + R, type control, and hit Enter.
2. Go to **System and Security > Windows Defender Firewall**.
3. On the left, click **“Turn Windows Defender Firewall on or off”**.
4. Make sure **all 3 profiles** are turned **ON**:Domain network,Private network,Public network

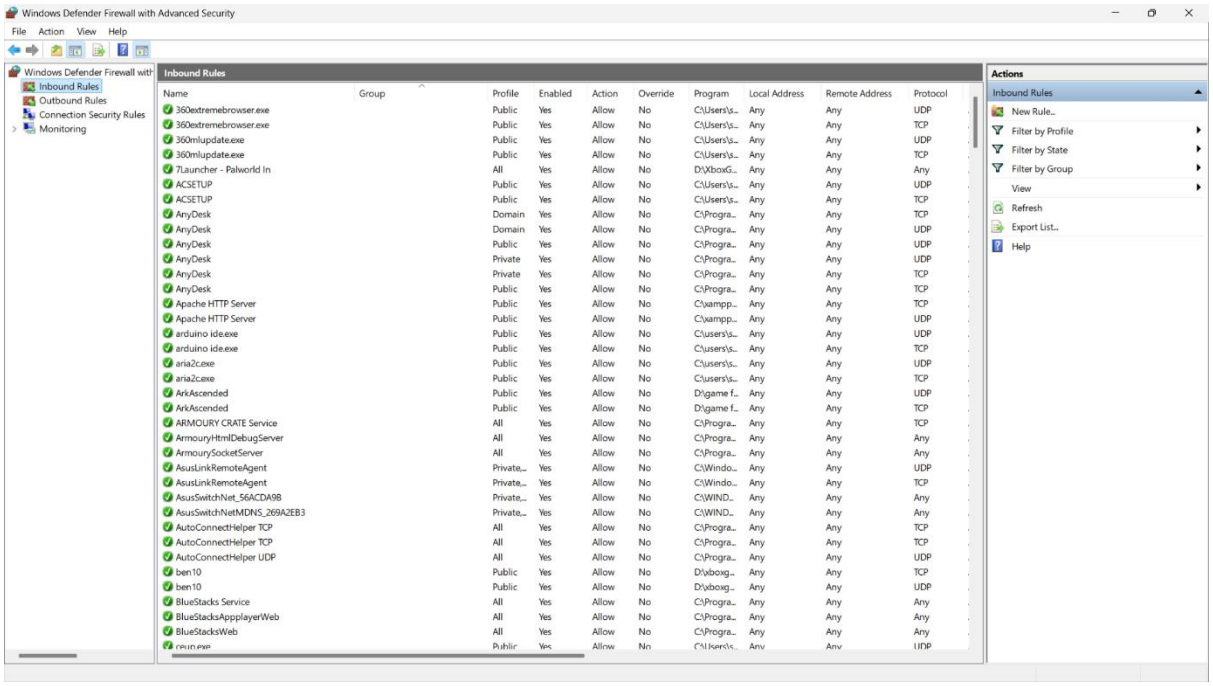
Screenshot :



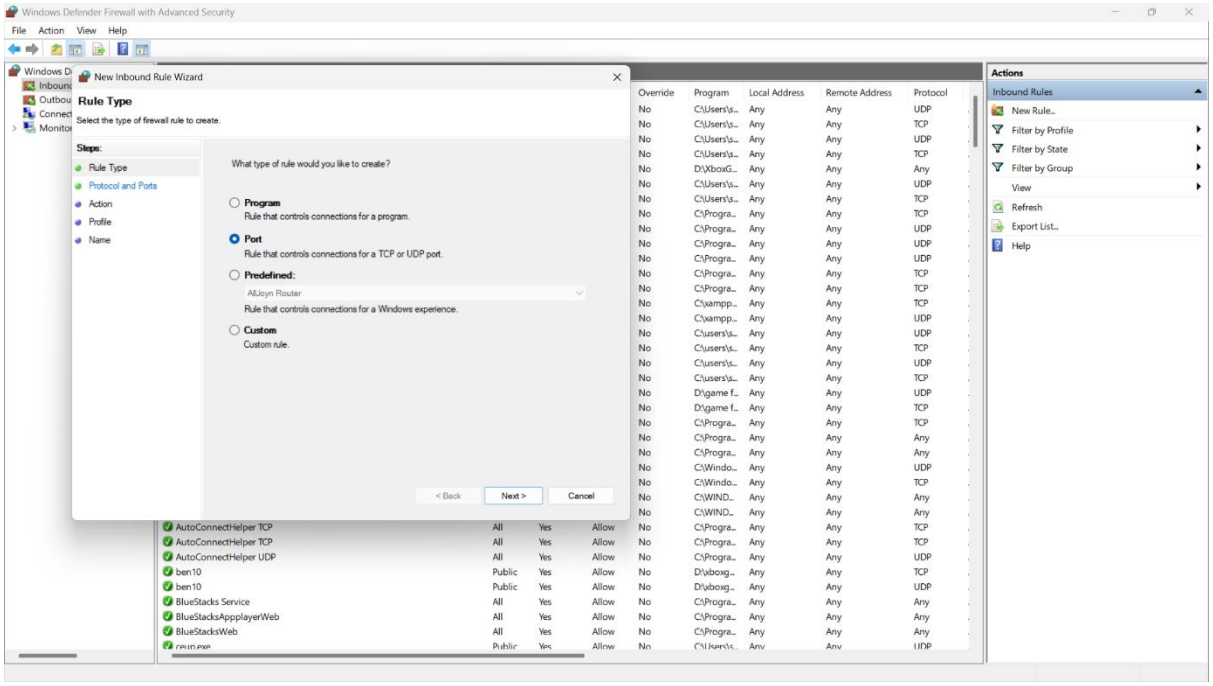
2. Create Custom Firewall Rule to Block a Port (e.g., Port 23 for Telnet)

1. Open **Windows Defender Firewall with Advanced Security**:
    - Press Windows + R, type wf.msc, press Enter.

Screenshot :

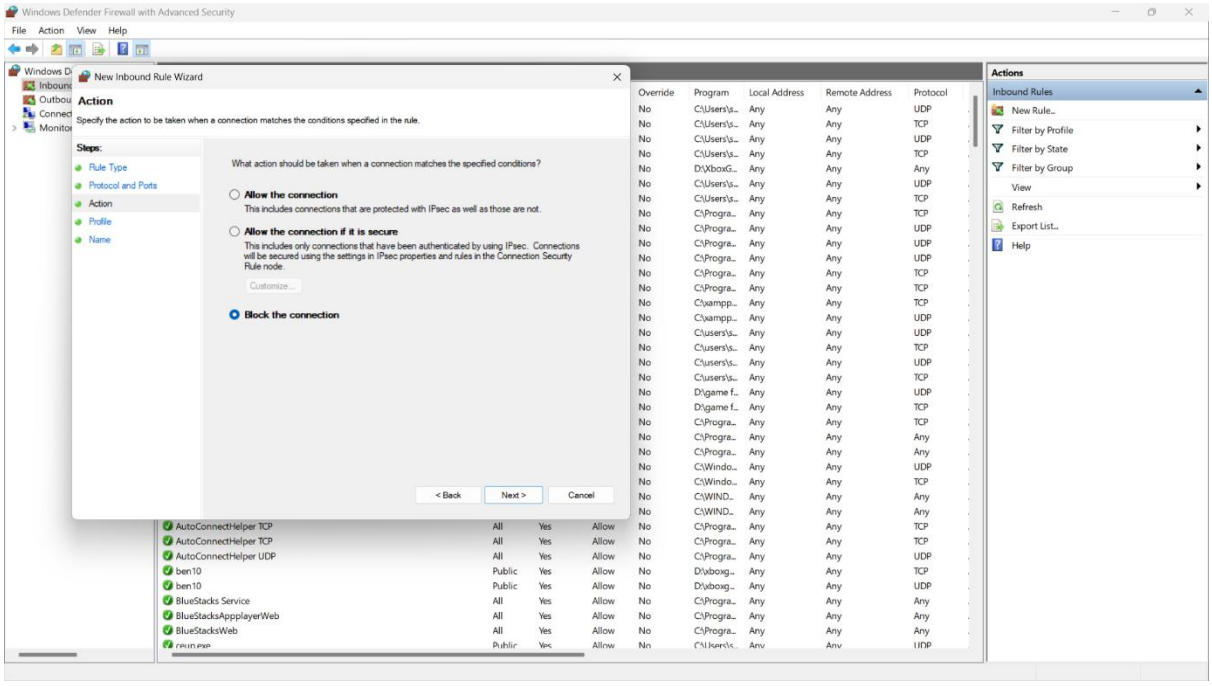


In the left pane, click **Inbound Rules** > then on the right, click **New Rule**.

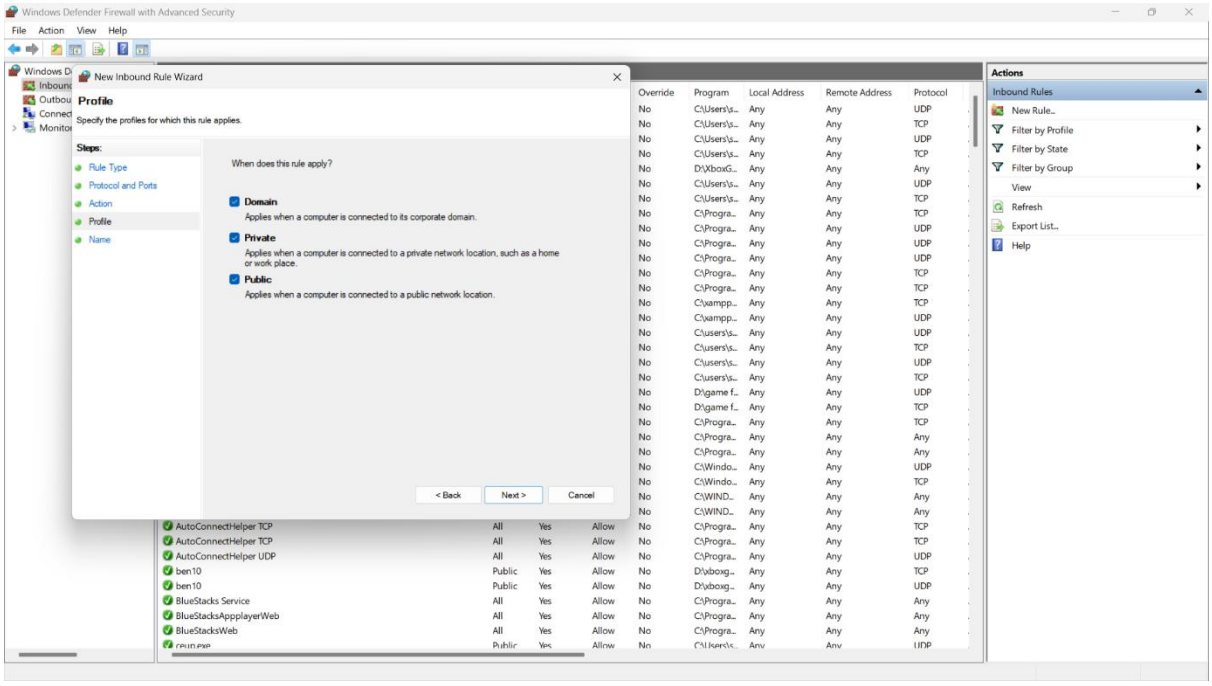


Select **Port**, click **Next**.

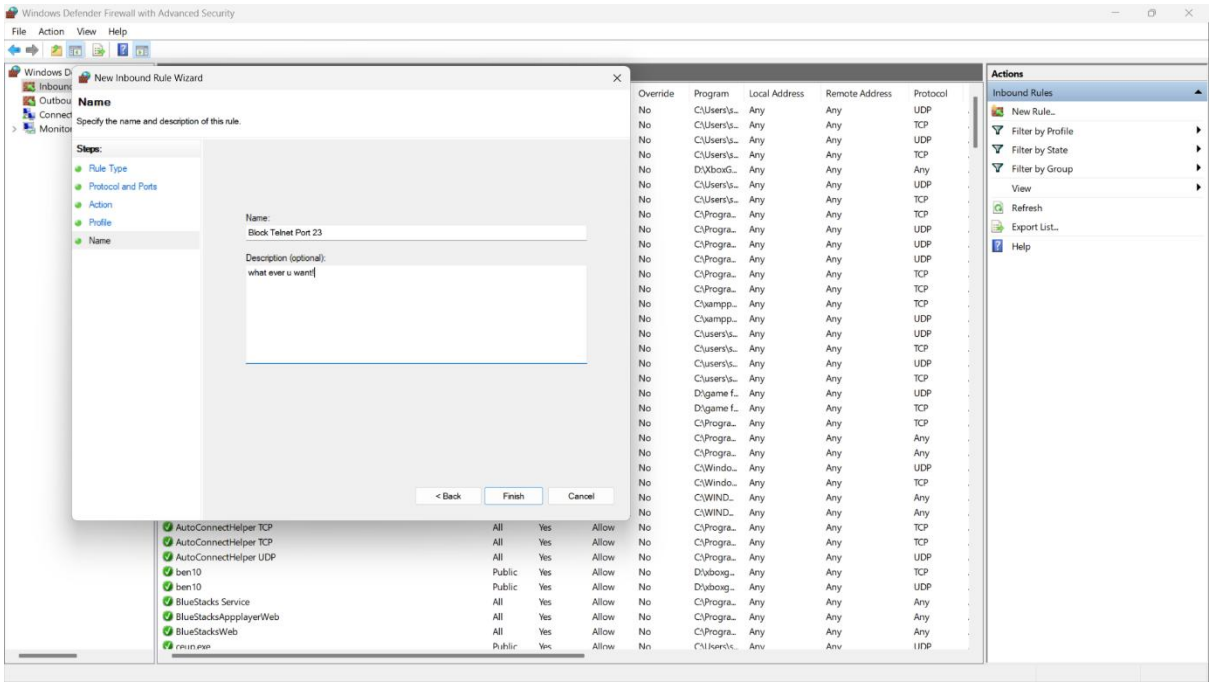
Choose **TCP**, enter 23 (for Telnet).



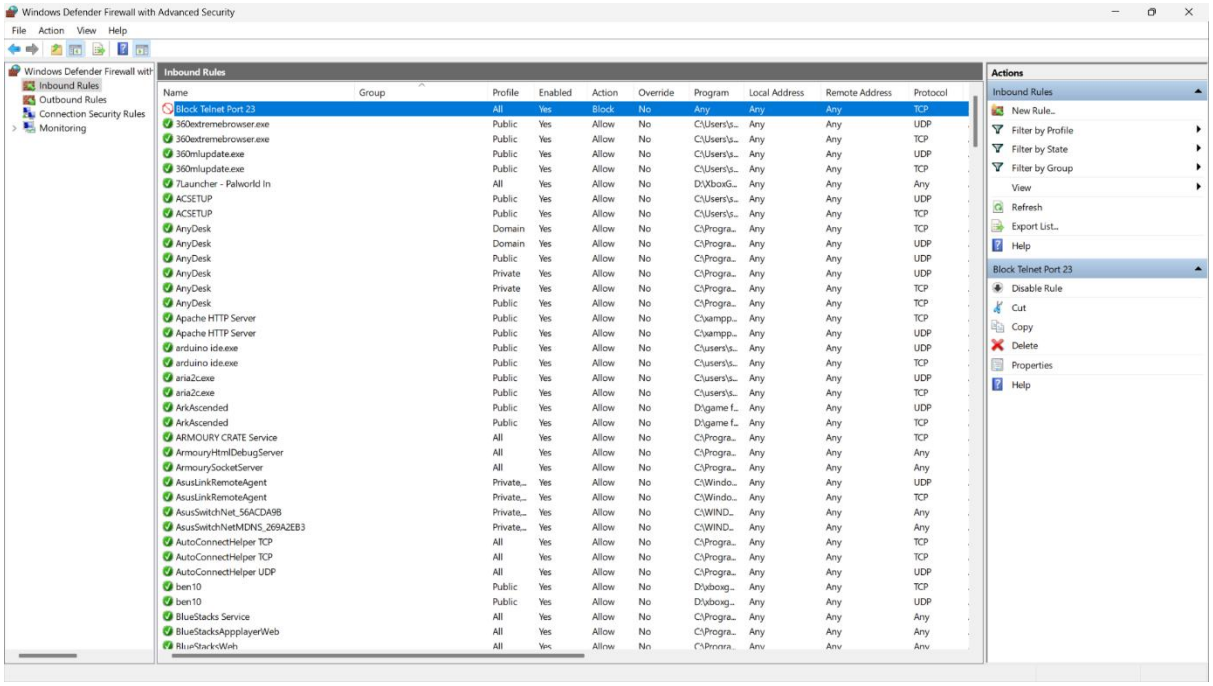
Select **Block the connection**, click **Next**.



Apply it to **Domain, Private, Public** > click **Next**.



Name it like: Block Telnet Port 23 and finish.



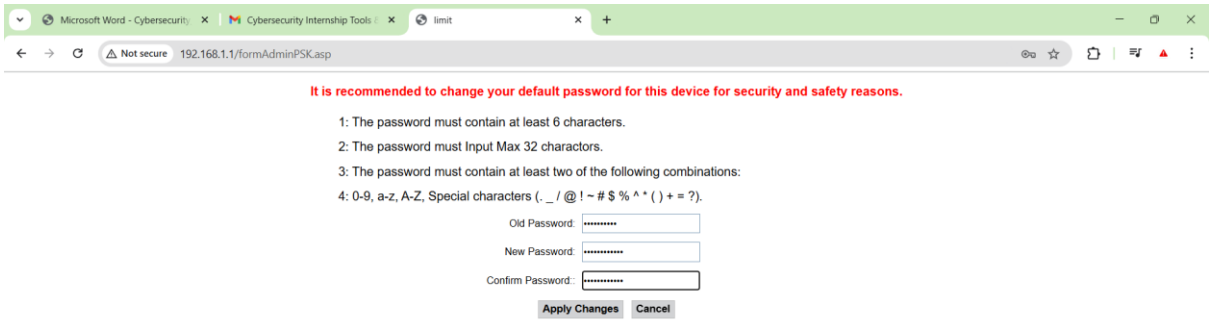
3. Change Default Router Admin Password

1. Open a web browser and type your router's IP (commonly 192.168.0.1 or 192.168.1.1).

Screenshot :

Login using default credentials.

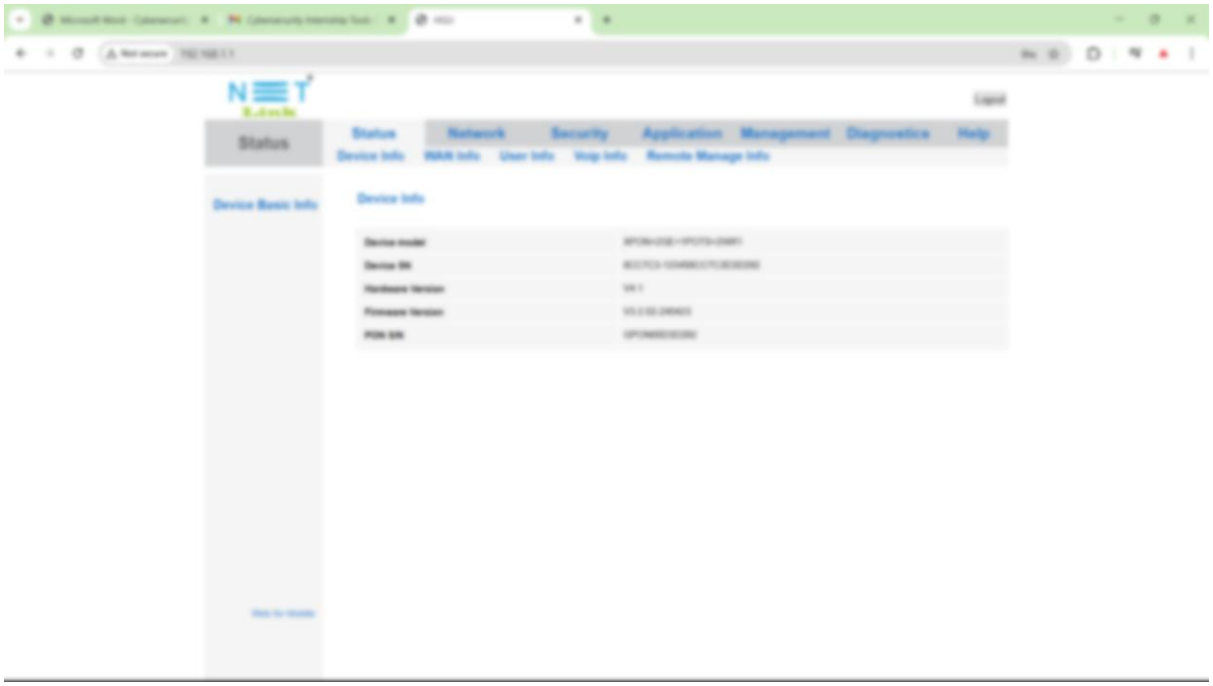
Go to the **Admin settings** or **Maintenance tab**.



Find the **password field** and change to a **strong password** (e.g., 12+ characters, mix of letters, numbers, symbols).

Save settings.

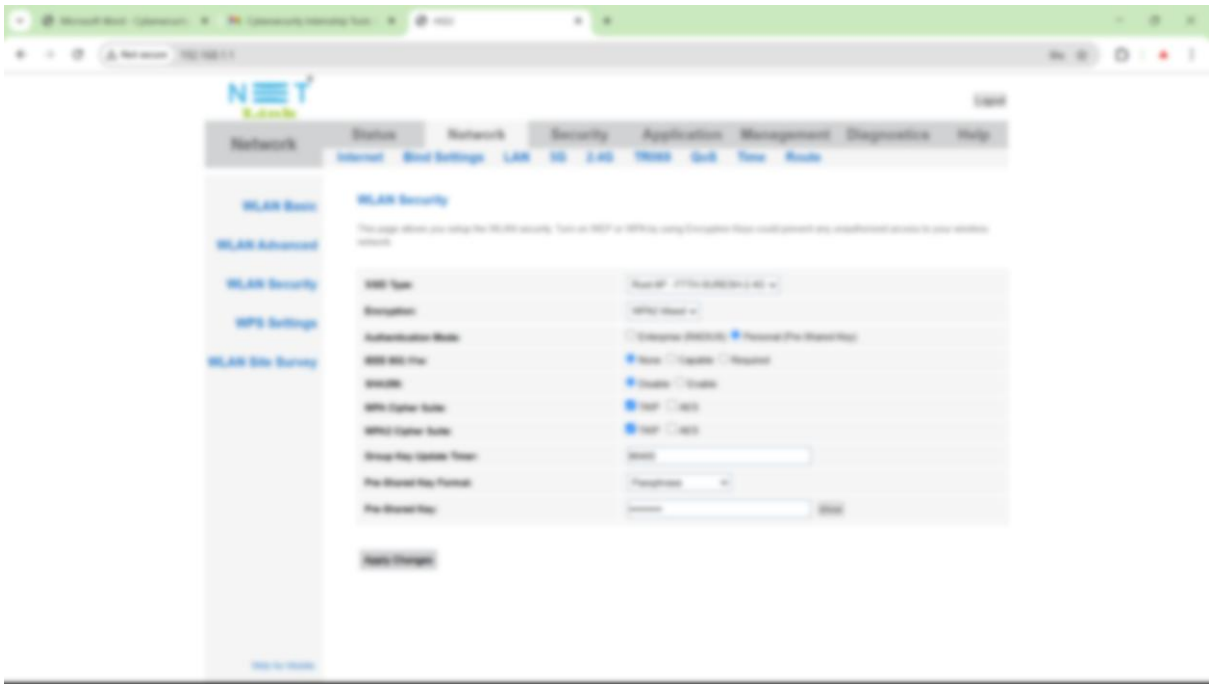
After successfully change the password then login with credientials.

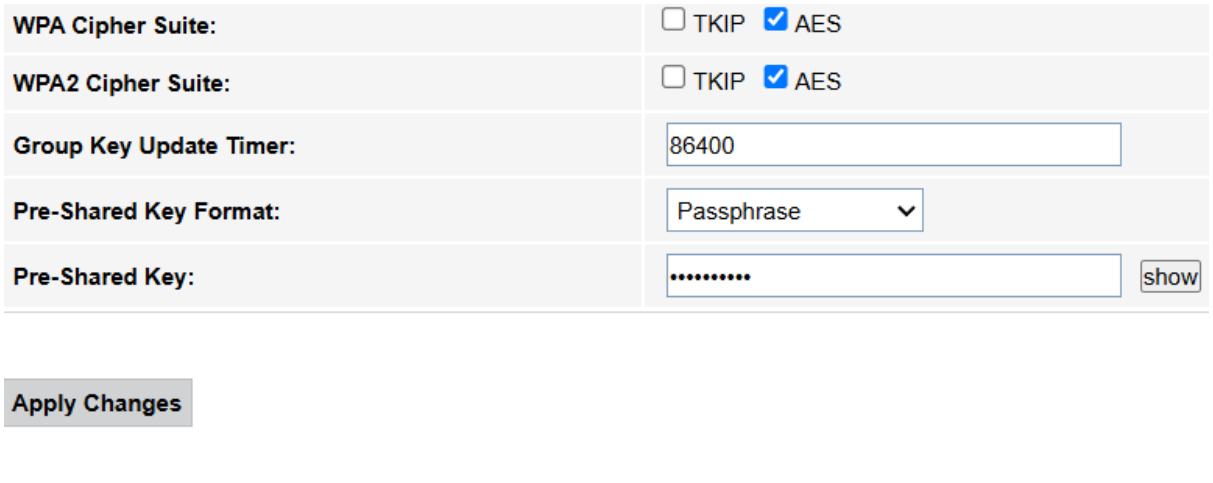


4. Enable WPA2/WPA3 Encryption for Wi-Fi

1. In the router settings (same as above), find **Wireless Settings > Security**.
2. Under **Security Mode**, choose:
    - **WPA2-PSK** or **WPA3-PSK** (if available).
3. Set a **strong Wi-Fi password** (different from router admin).
4. Save the changes and reconnect your devices.

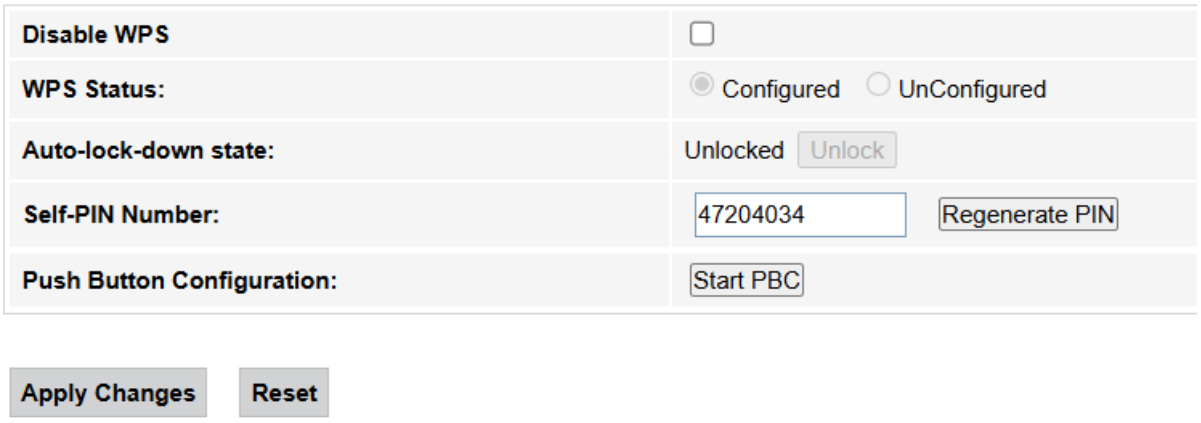
Screenshot :





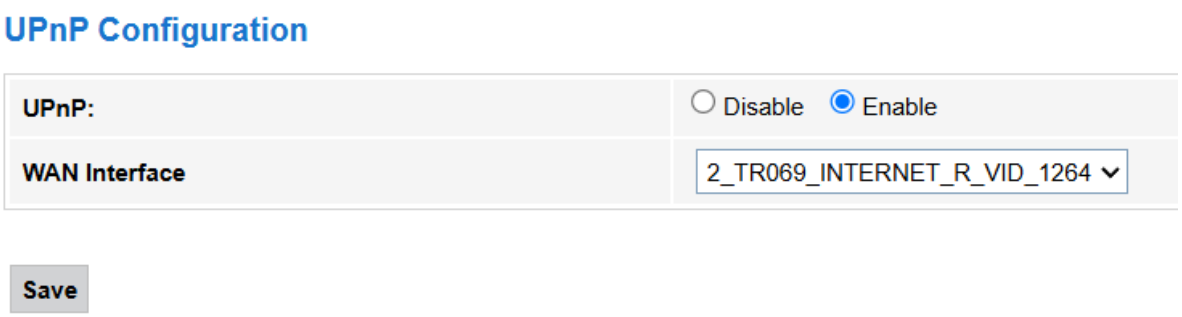
5. Disable WPS and UPnP

1. In router settings, look under **Advanced Settings** or **Security > WPS Settings**.
2. **Turn off WPS** (Wi-Fi Protected Setup).



3. Go to **Advanced > UPnP Settings**.
4. **Disable UPnP** (Universal Plug and Play).
5. Save settings.

Screenshot :



3. Wireshark Traffic Analysis

I used Wireshark to capture and analyze network traffic. The following traffic types were observed:

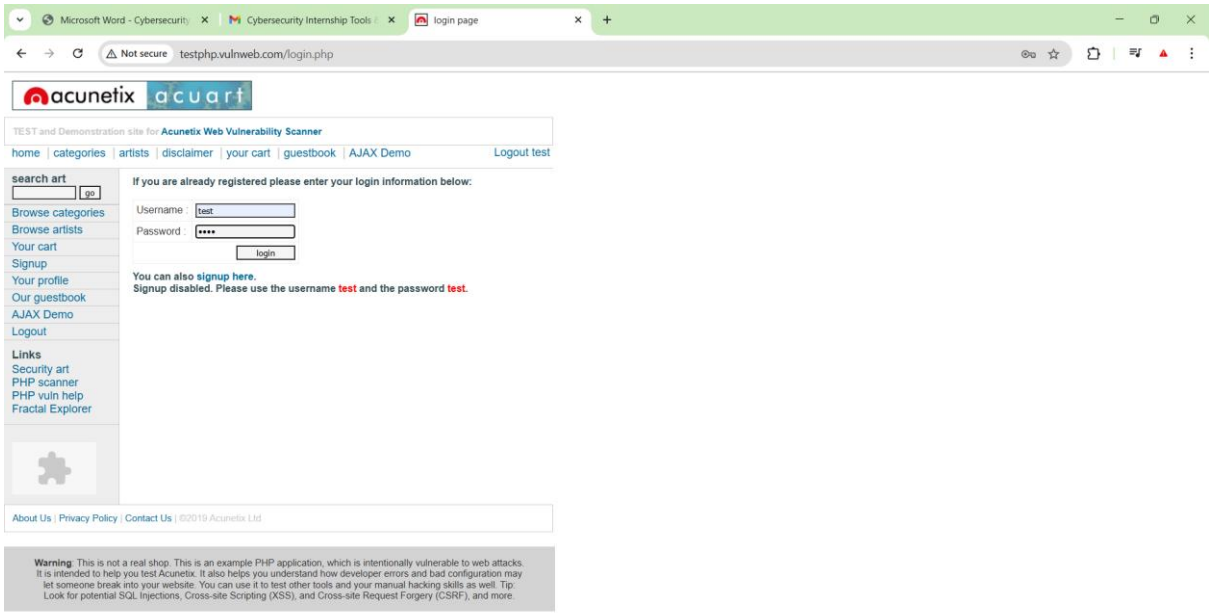
HTTP: Clear-text web browsing data between client and web server.
DNS: Domain name lookups performed by applications to resolve domain names to IP addresses.
ICMP: Ping requests and responses used to test network connectivity.

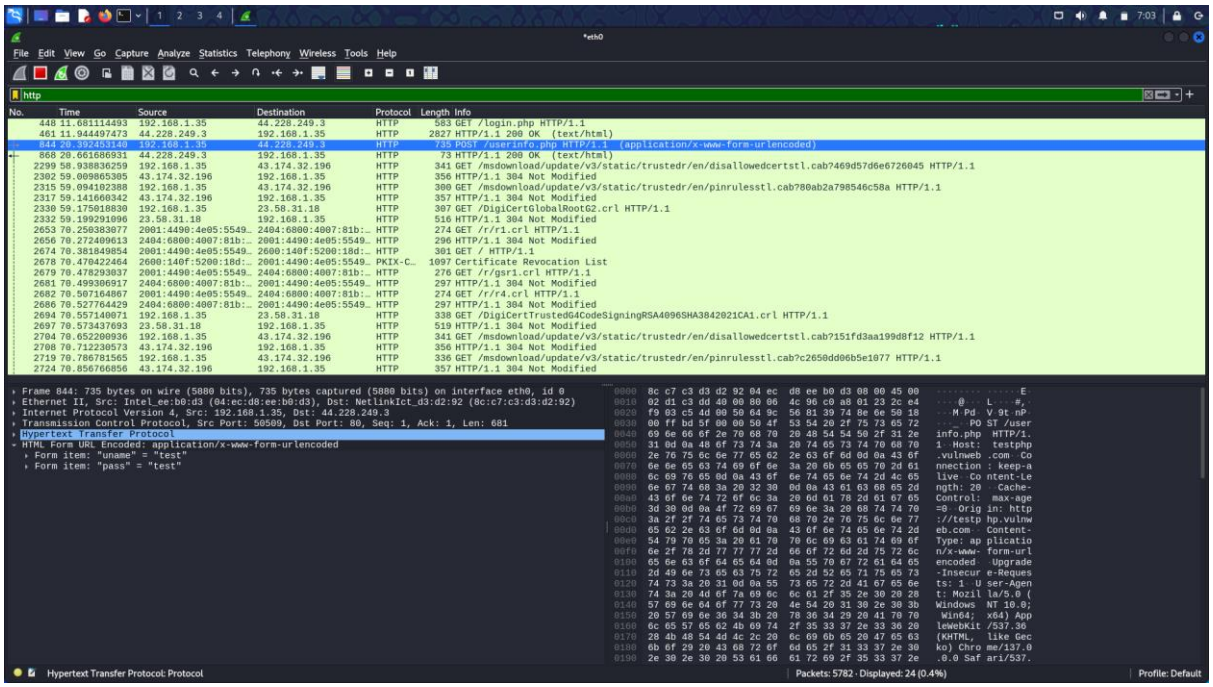
I also identified packets from unusual IP addresses making repeated connections, which could indicate potential scanning or bot activity.

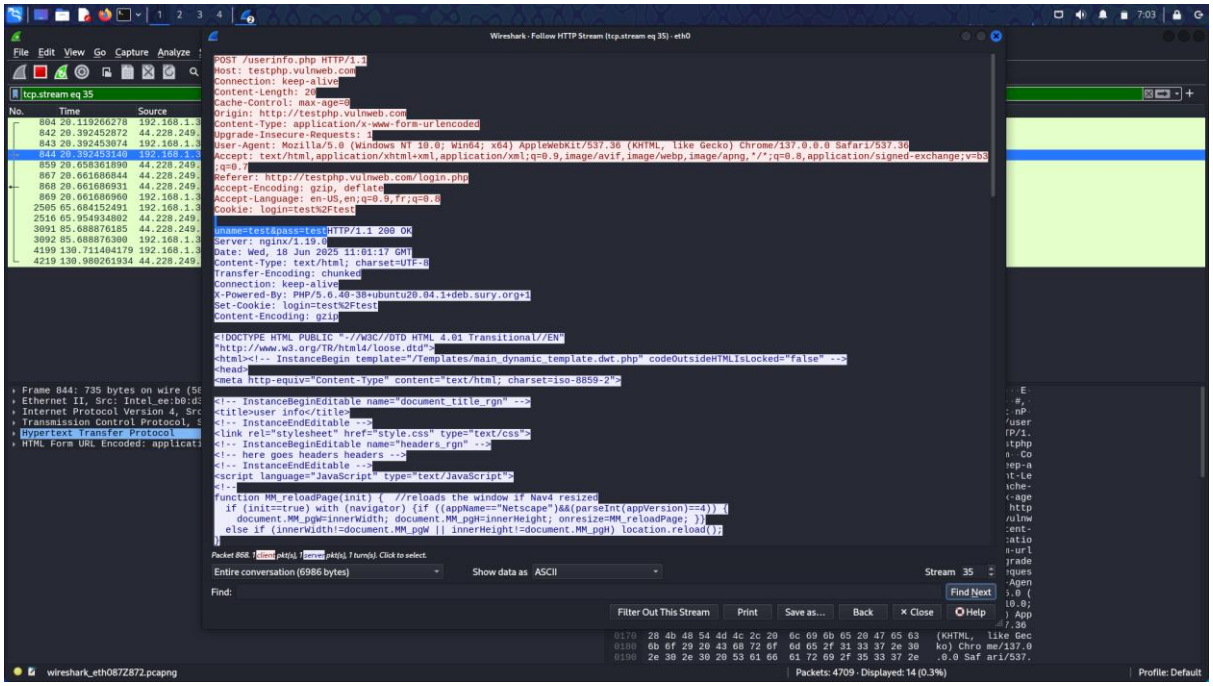
Observe Common Traffic Types

- **HTTP**
  - You'll see requests to websites, headers, methods like GET or POST
  - Try visiting a non-HTTPS site to generate visible HTTP traffic

Screenshot :

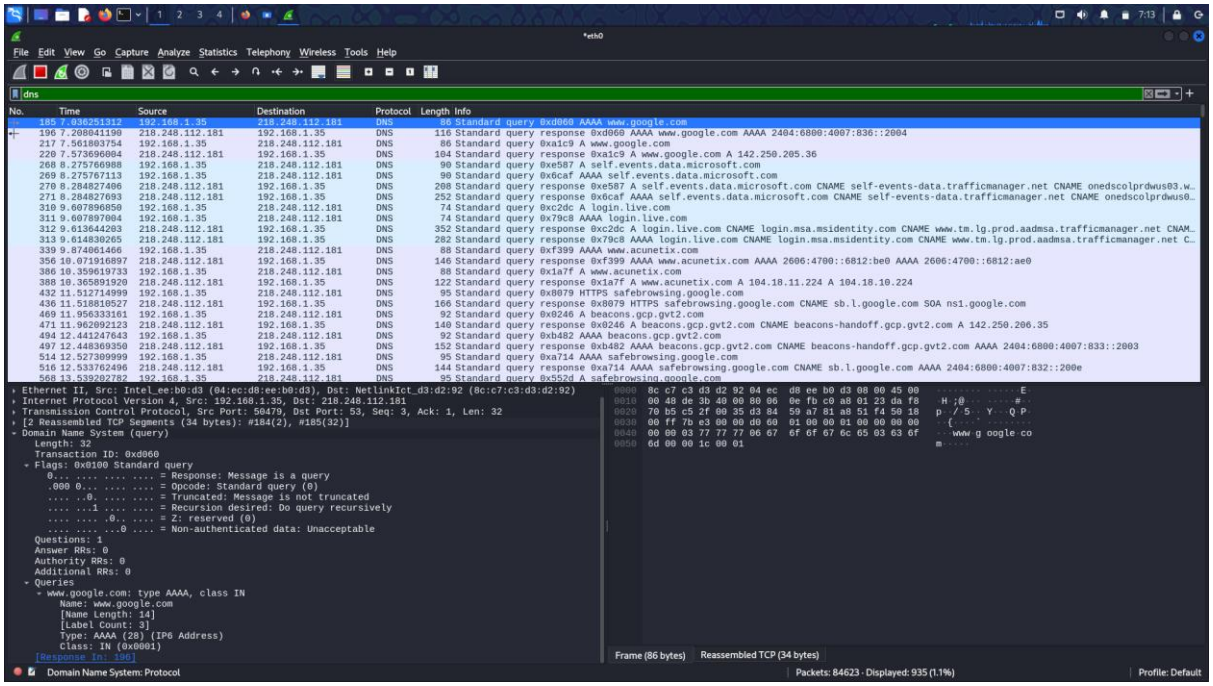






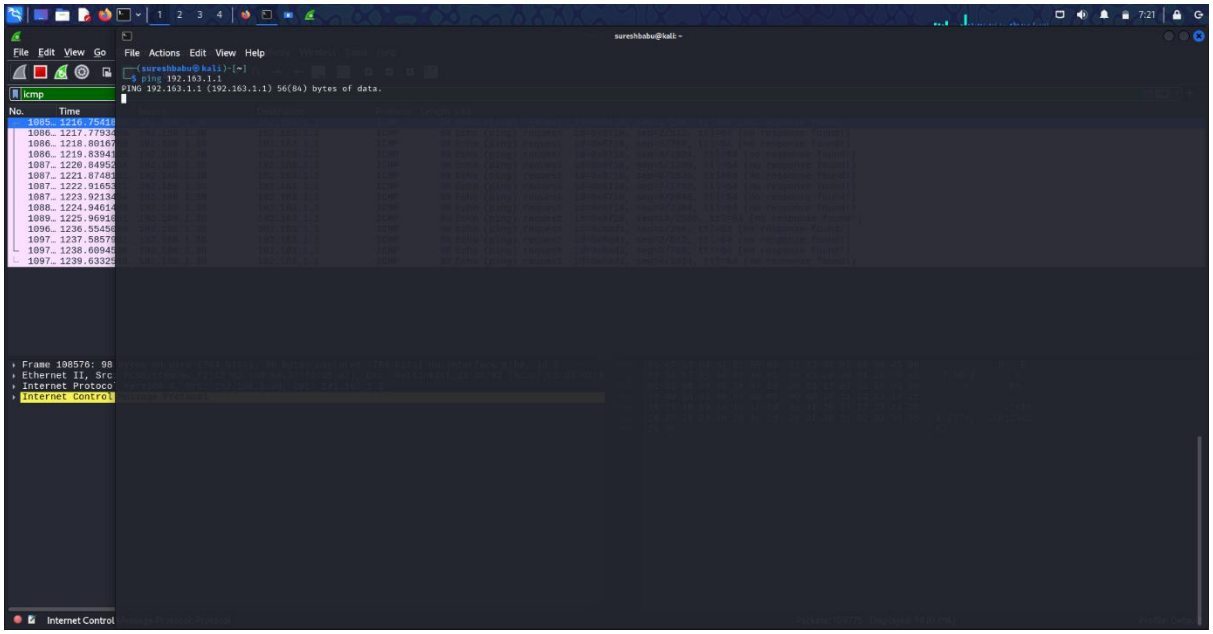
- **DNS**
  - Filter: dns
  - Shows domain name queries like www.google.com → IP

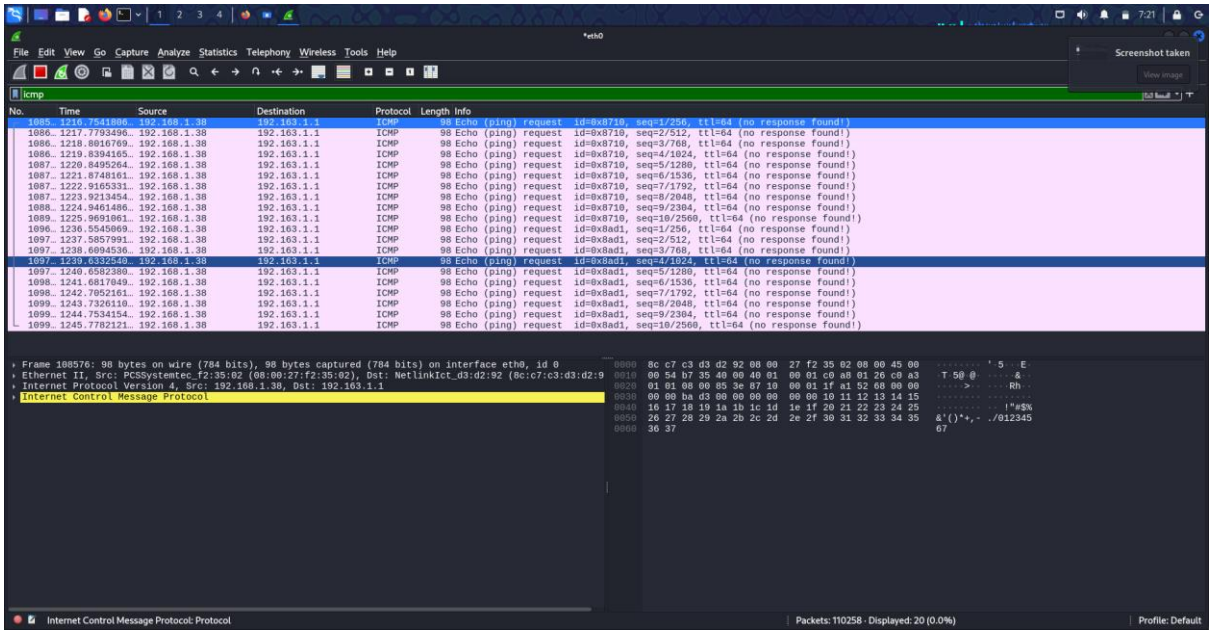
Screenshot :



- **ICMP**
  - Filter: icmp
  - Run ping google.com in Command Prompt to generate

Screenshot :





Identify Unusual Traffic (Threat Indications)

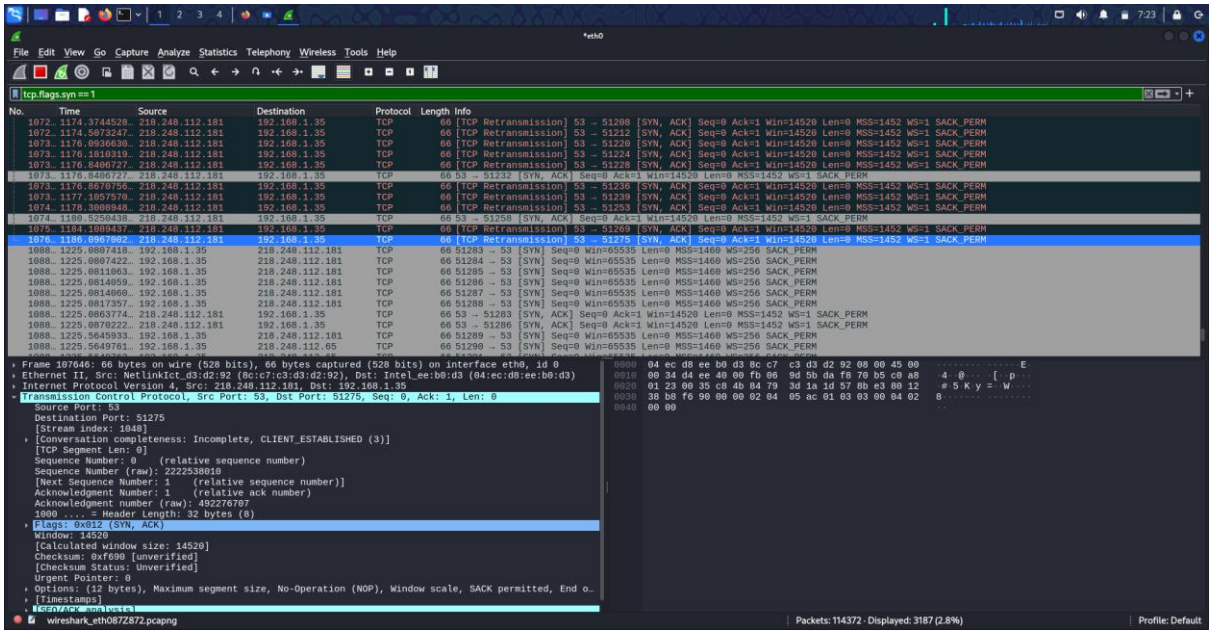
Look for Suspicious Activity:

- IPs outside your subnet making repeated connections
- High frequency of SYN packets (tcp.flags.syn == 1)
- Strange ports or unknown protocols

Filter example for TCP SYN floods:

tcp.flags.syn == 1 && tcp.flags.ack == 0

Screenshot :



4. Extended Security Measures & Awareness

For Larger or Enterprise Networks

To strengthen security in enterprise or more complex environments, I recommend implementing:

- **Intrusion Detection Systems (IDS):** Detect and alert on suspicious or malicious network activity.
- **VLANs (Virtual LANs):** Segment networks to isolate traffic and limit the spread of threats.

- **Access Control Policies:** Enforce least-privilege access to restrict user permissions.
- **VPNs (Virtual Private Networks):** Secure remote access by encrypting traffic.

User Awareness & Cyber Hygiene

Educating end-users helps reduce social engineering and human-error-based threats:

- Recognize phishing attempts, suspicious attachments, and spoofed websites.
- Regularly update all software and firmware to patch vulnerabilities.
- Secure home or office Wi-Fi with strong passwords and WPA2/WPA3 encryption.

5. Conclusion

By applying basic network security configurations and using Wireshark for traffic analysis, I learned how to identify potential threats and safeguard a network environment. These foundational steps are vital in building a secure network and can scale effectively to larger, enterprise-grade setups.

Task 2: Introduction to Web Application Security using OWASP ZAP and WebGoat

1. Introduction

This report documents the identification and exploitation of three common web application vulnerabilities using OWASP ZAP against WebGoat, an intentionally vulnerable application created for security training. The primary goal is to understand how vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF) can be discovered and exploited, and how they can be mitigated.

2. Environment Setup

Component	**Details**
Vulnerable App	**WebGoat (Docker)**

Component	Details
Scanning Tool	**OWASP ZAP**
URL	**http://localhost:8081/WebGoat**
Authentication Used	**guest / guest**
Browser Proxy	**ZAP Browser for interception**

3. Vulnerability Details

A.SQL Injection

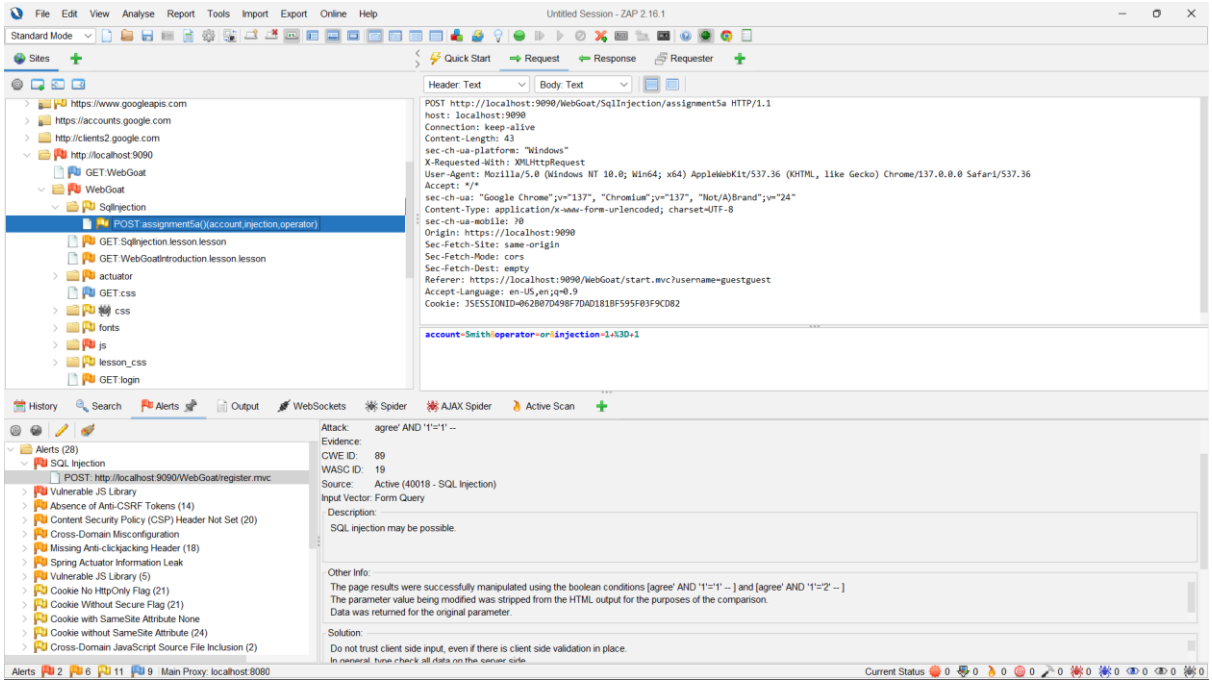
- Module: WebGoat → SQL Injection (Intro)
- Description: SQL Injection occurs when untrusted input is inserted directly into a SQL query without sanitization, allowing attackers to modify or leak database data.
- Payload Used:
- ' OR '1'='1
- Observation:
  The payload bypassed login logic and returned user records, confirming the injection.
- Exploitation Steps:
  1. Navigated to SQL Injection lesson.
  2. Entered ' OR '1'='1 in the login field.
  3. Successfully accessed unauthorized user data.
- Impact:
  Can be used to bypass authentication or exfiltrate sensitive data.

Screenshot:

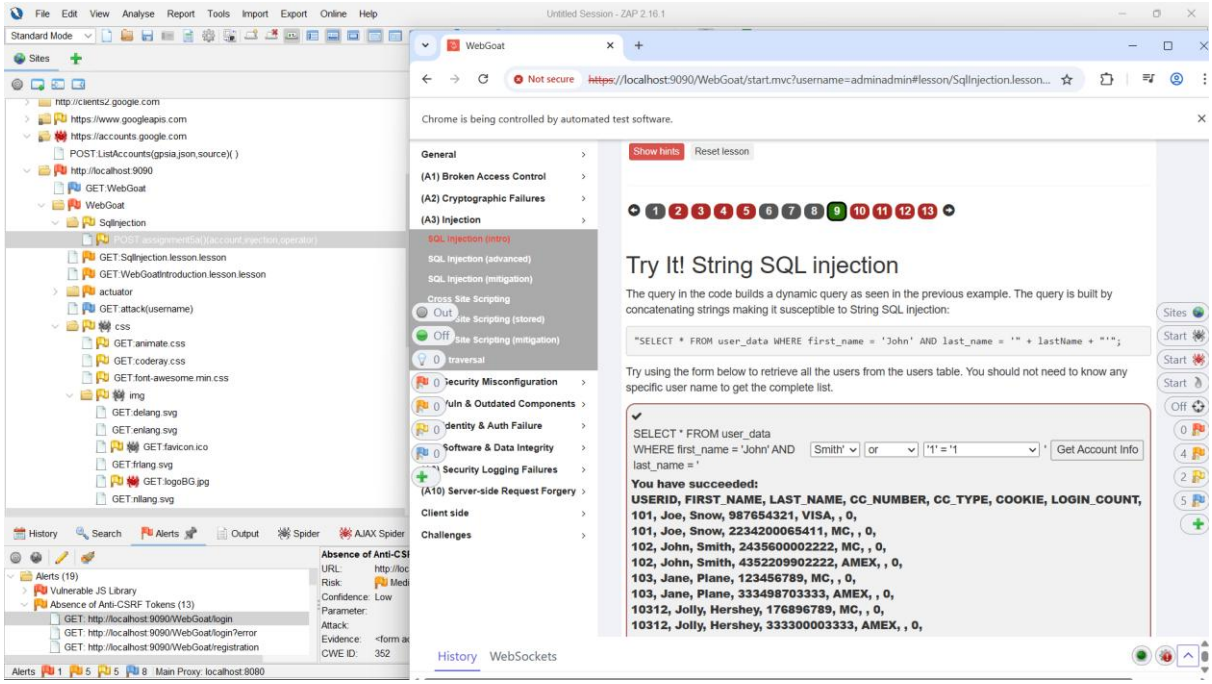
Actually the above mentioned sql injection is error-based injection that easily refleted on login page then we move into unknown user login page.

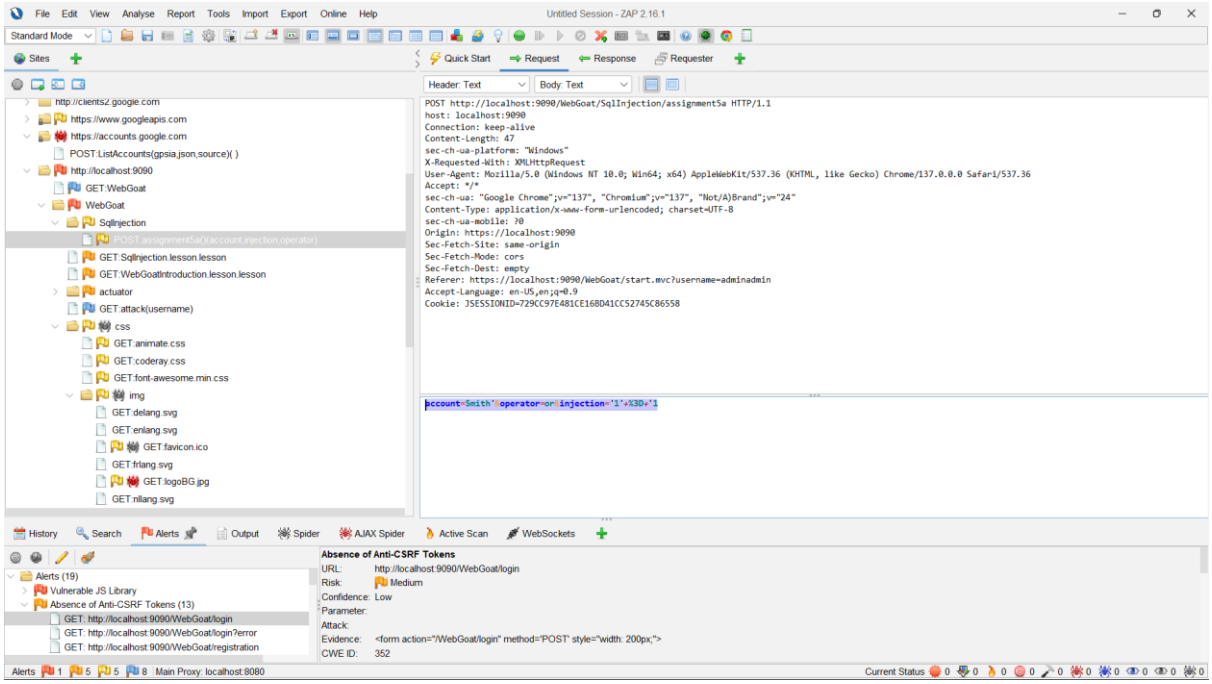
When this error-based sql injection tested and work in admin login page ,definitely login with root access panel.

First login on webgoat then move into sqli lesson and inject your payload then capture your request.

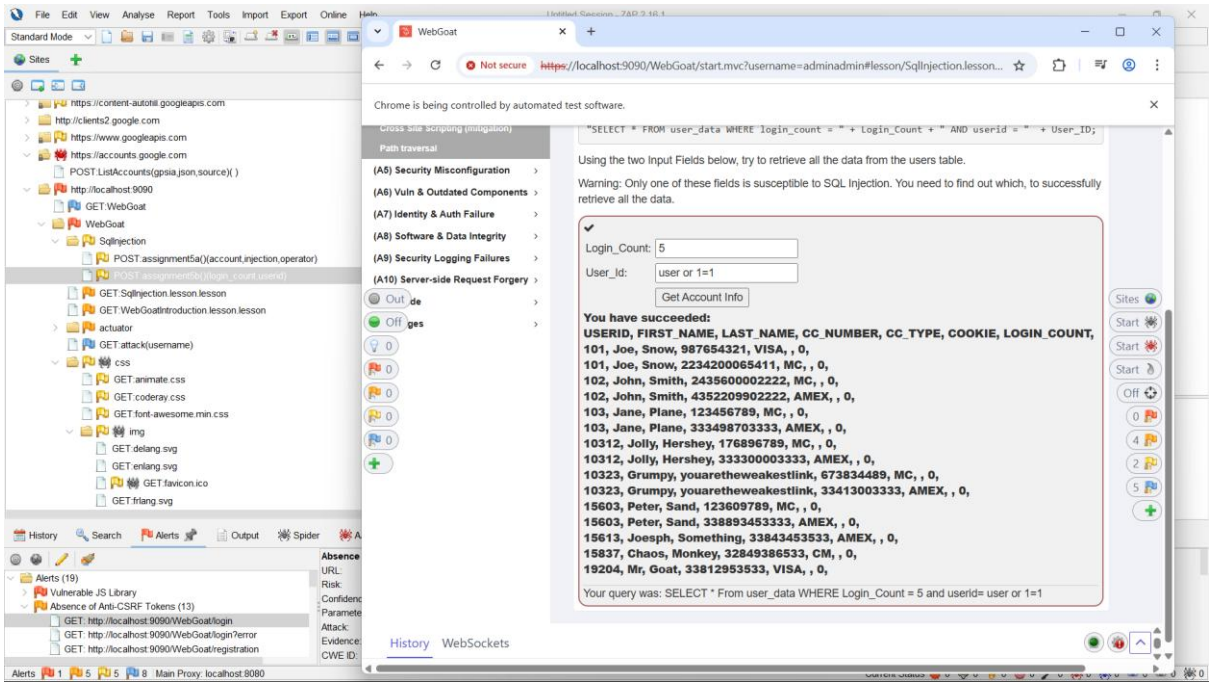


The request page we see the query processing and sqli work the respose will reflect on both response code and web page.

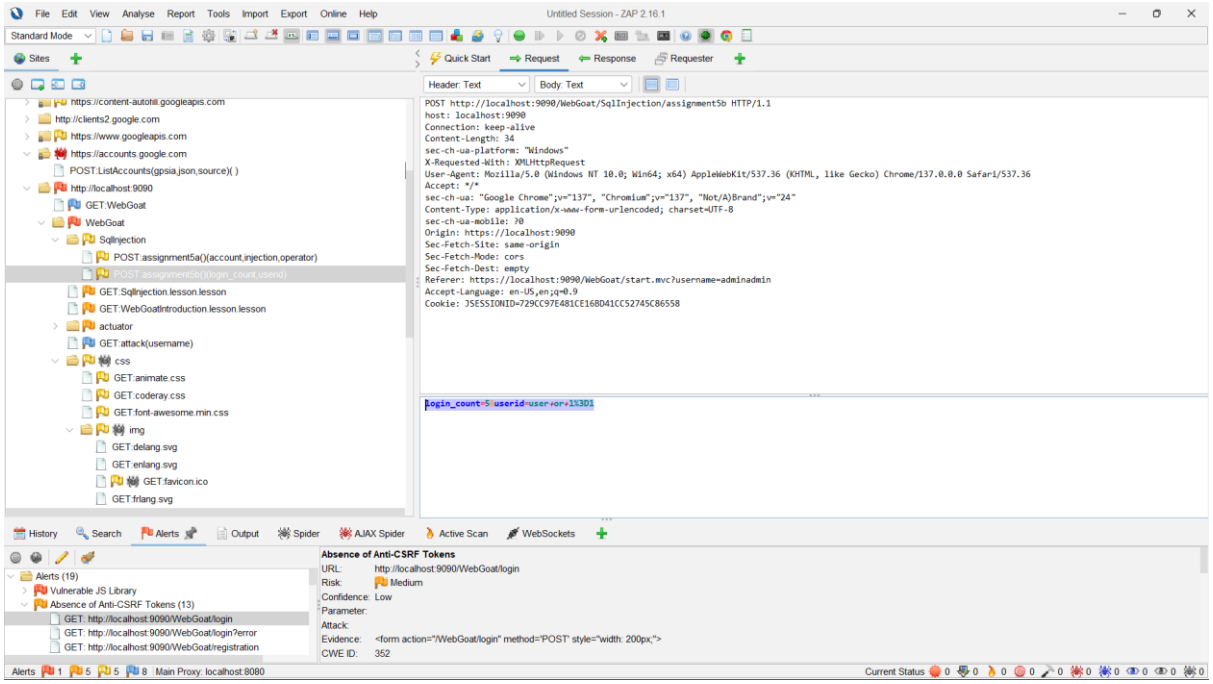




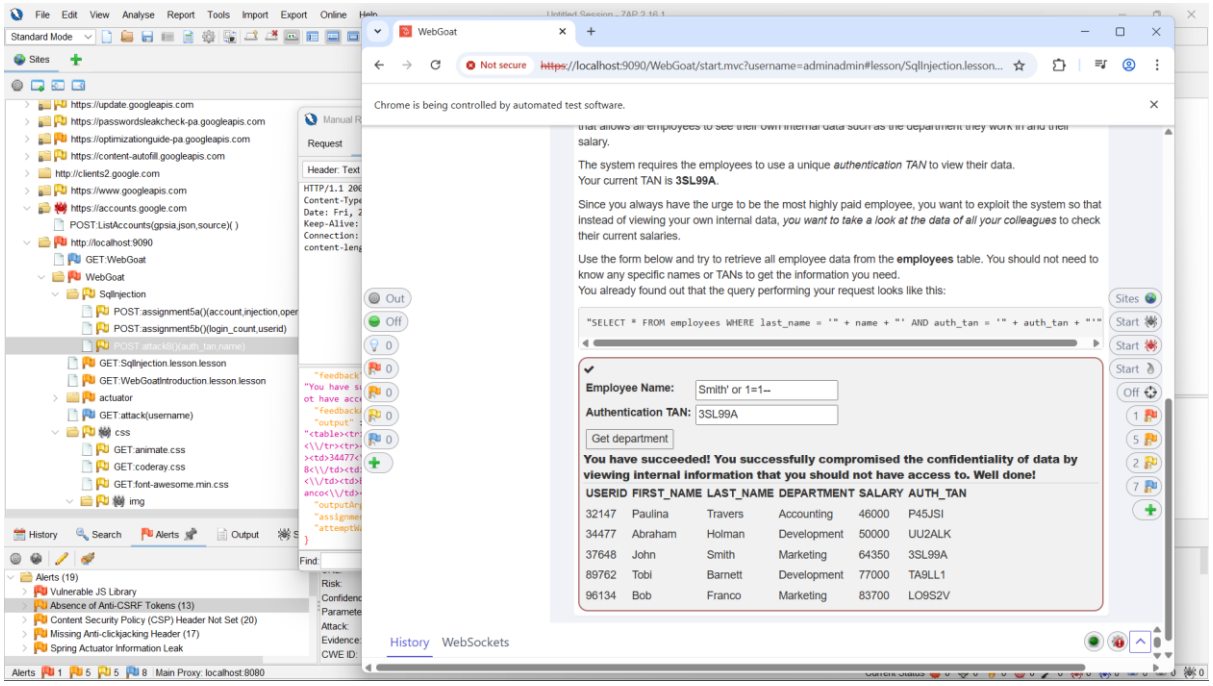
Third picture was shown the request of Smith' or '1'='1 and the response is driven in picture 2.



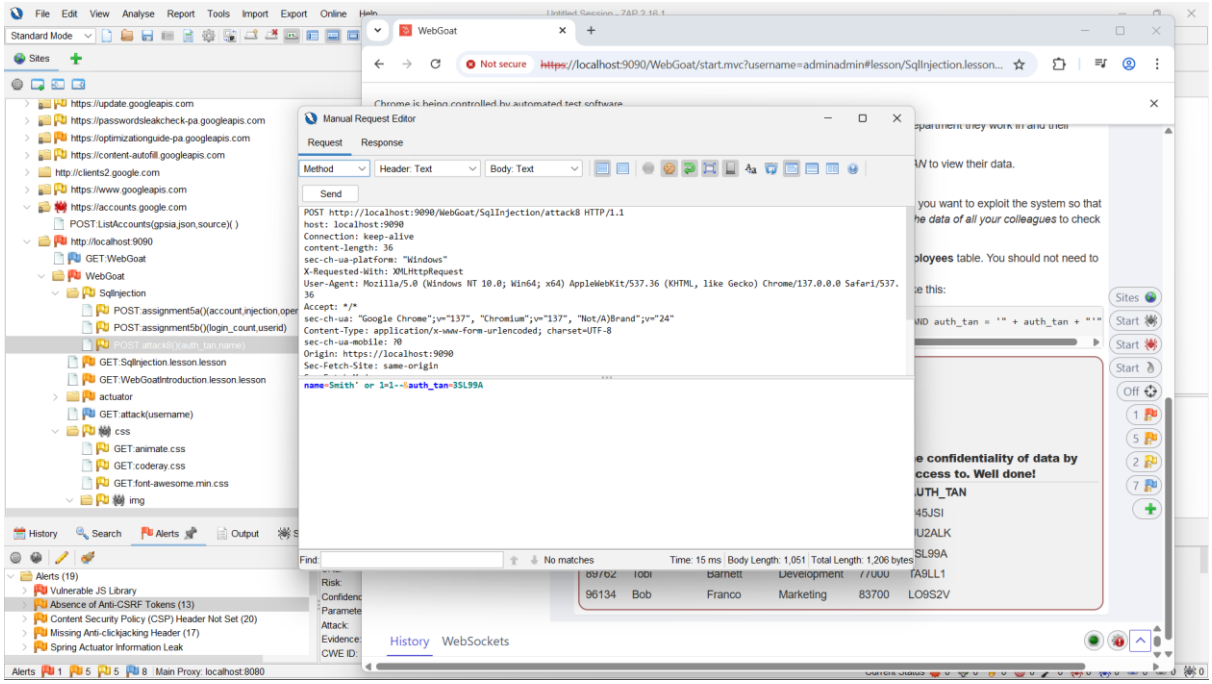
Now I use the same strategy on Login_count with user or 1=1 after I fetch all the user information in database.



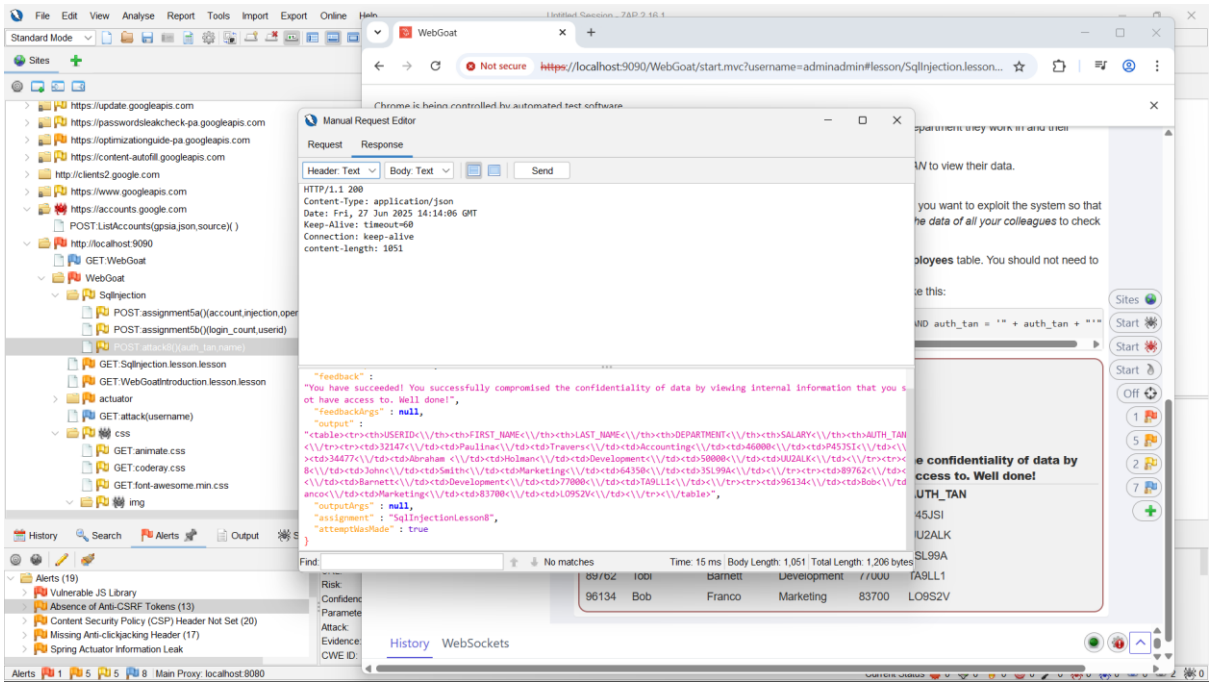
This picture was shown the request tab of user information sql query processing.



Post request is only get database information on sql injection attack and this is p2 or p3 attack level.



Another way of error-based attack end with – this and get information on response tab.



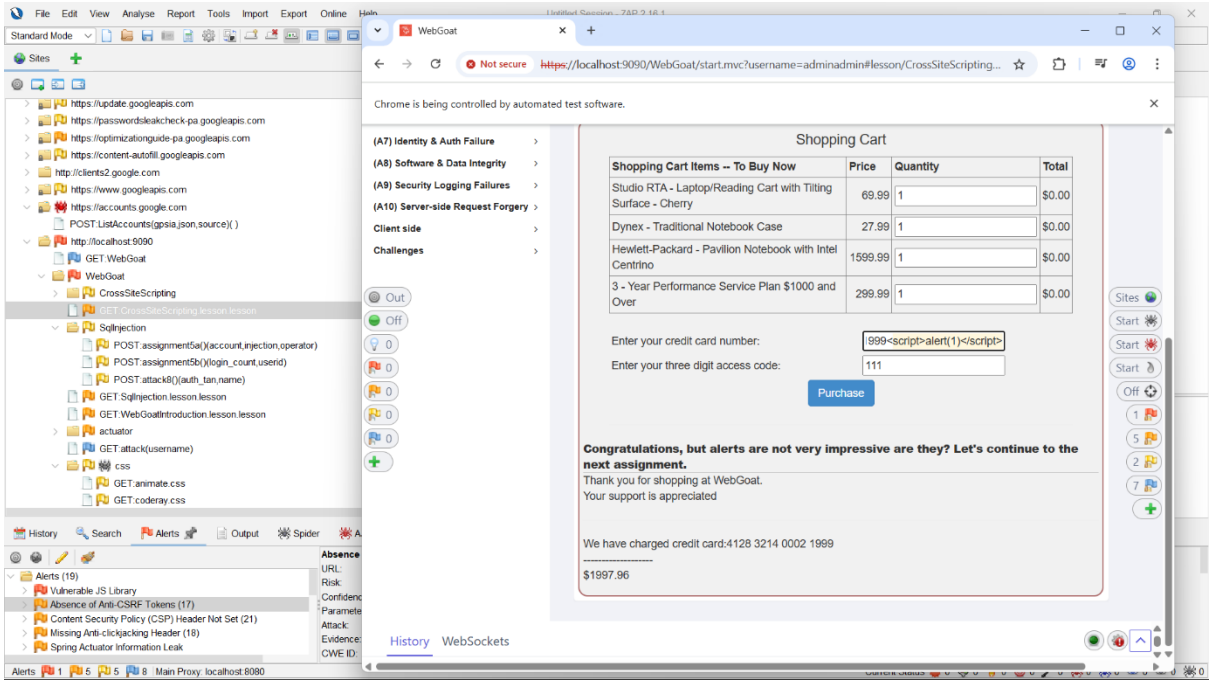
Mitigation:

- Use parameterized SQL queries (e.g., PreparedStatement in Java).
- Validate and sanitize all user input.
- Apply least privilege access to database accounts.

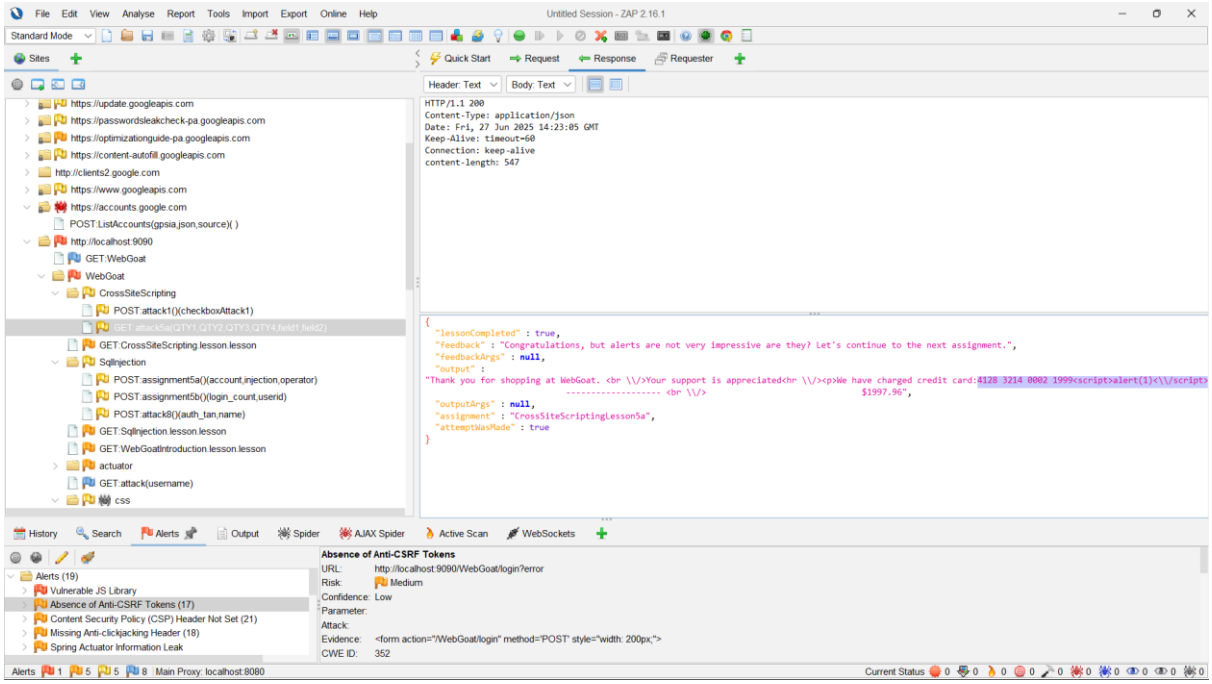
B. Cross-Site Scripting (XSS)

- **Module:** WebGoat → Cross-Site Scripting (Stored/Reflected)
- Description: XSS allows attackers to inject malicious scripts into web pages, which are then executed in users' browsers.
- Payload Used:
- <script>alert('XSS')</script>
- Observation:
  The JavaScript executed in the browser, confirming the vulnerability.
- Exploitation Steps:
  1. Submitted the above payload in the input field.
  2. Page rendered with an alert popup, demonstrating script execution.
- Impact:
  Could allow session hijacking, defacement, or redirection to malicious sites.

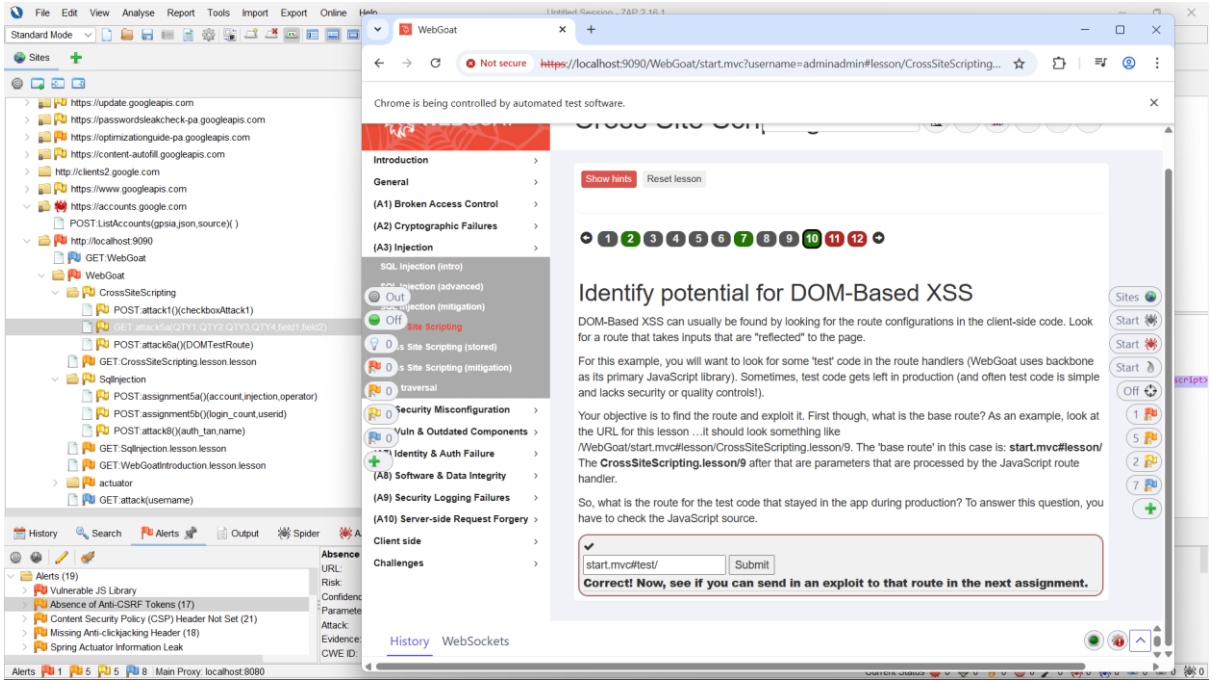
Screenshot:



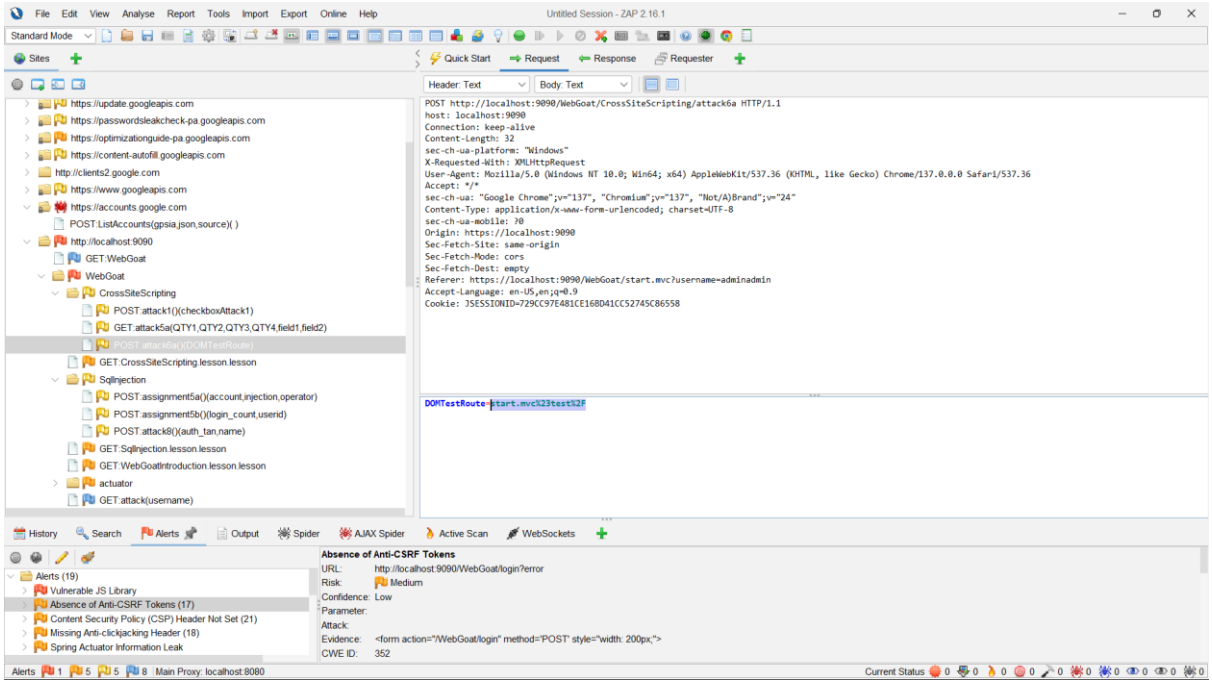
Fisrt i tested the cross-site scripting with the credit card number and I check it reflected.



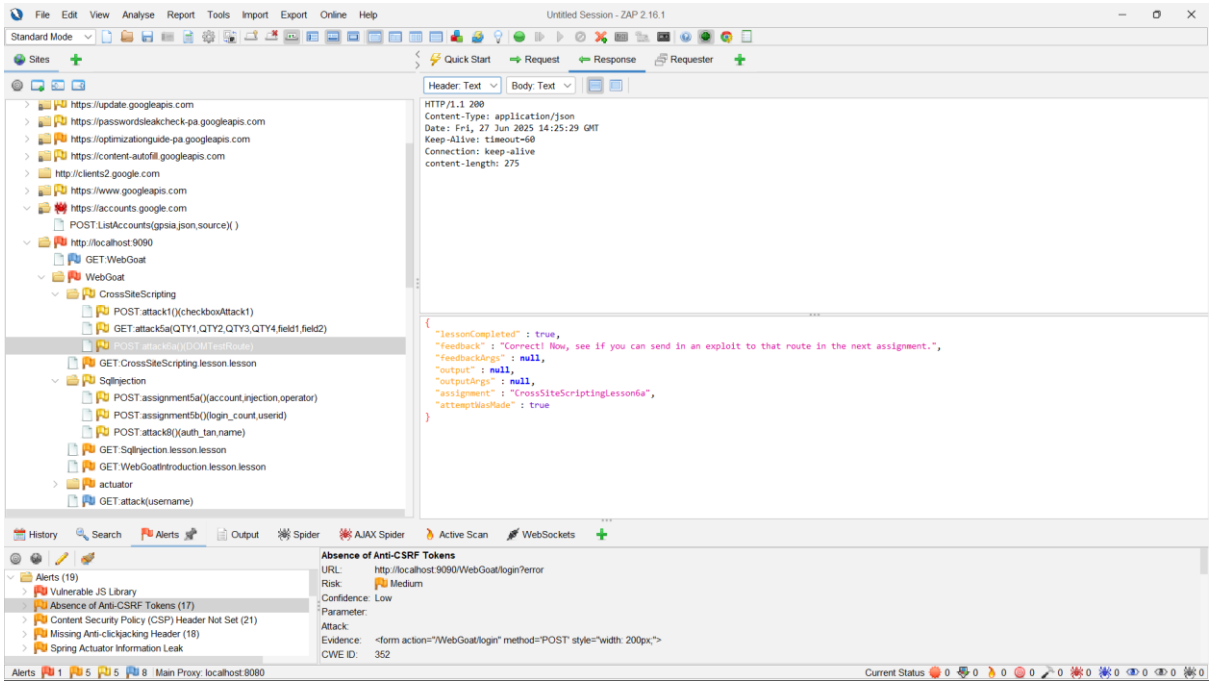
Fortunately, My test payload was work and I refleted on that page and response page give the response 200 ok..



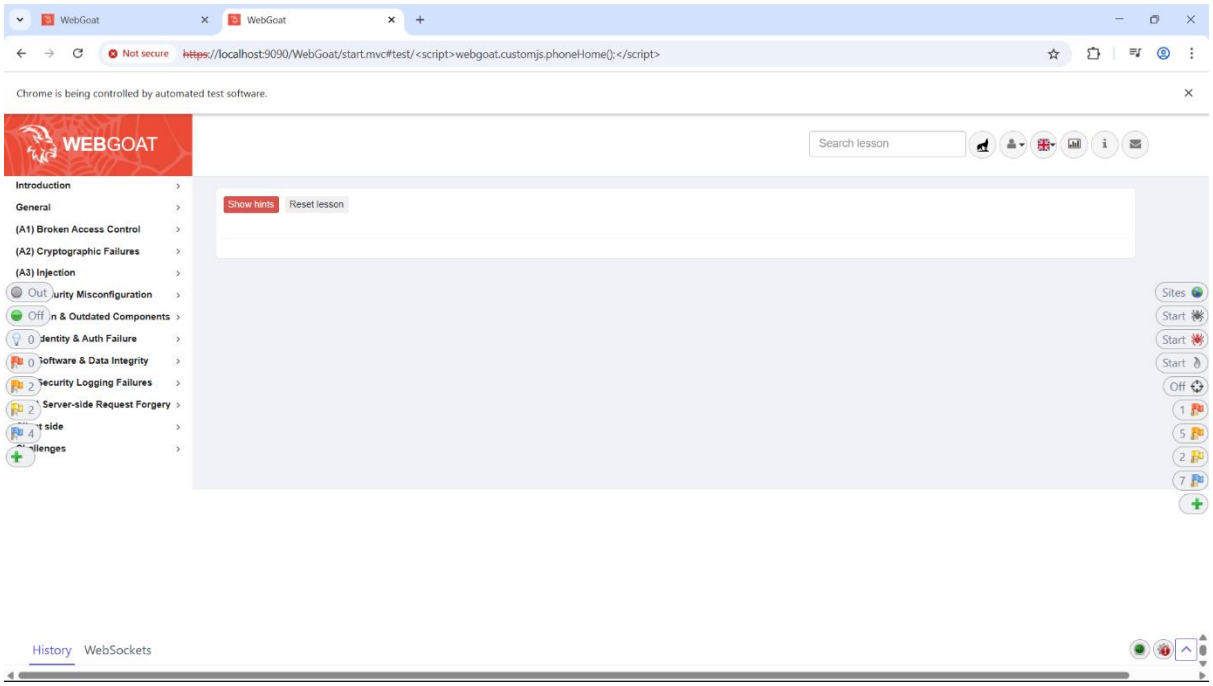
Next I tsted the DOM-based XSS on lesson 10 path=start.mvc#test/ here the path is correct for executing DOM.



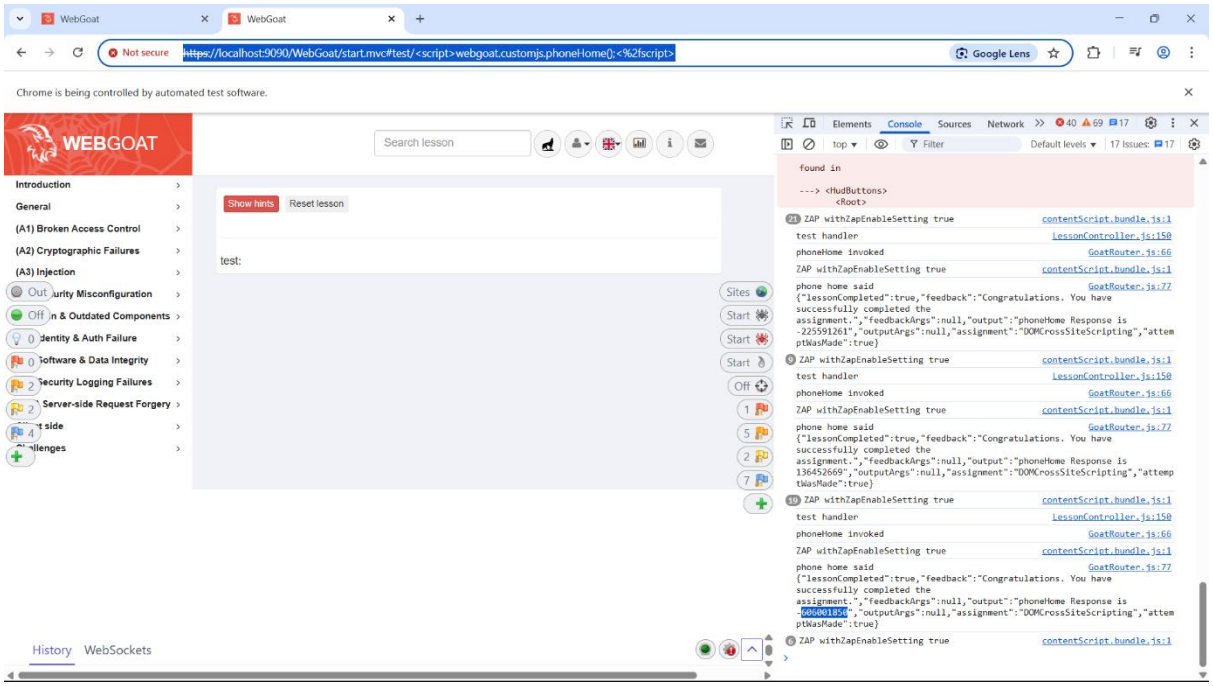
Here the requested query that clearly mentioned.



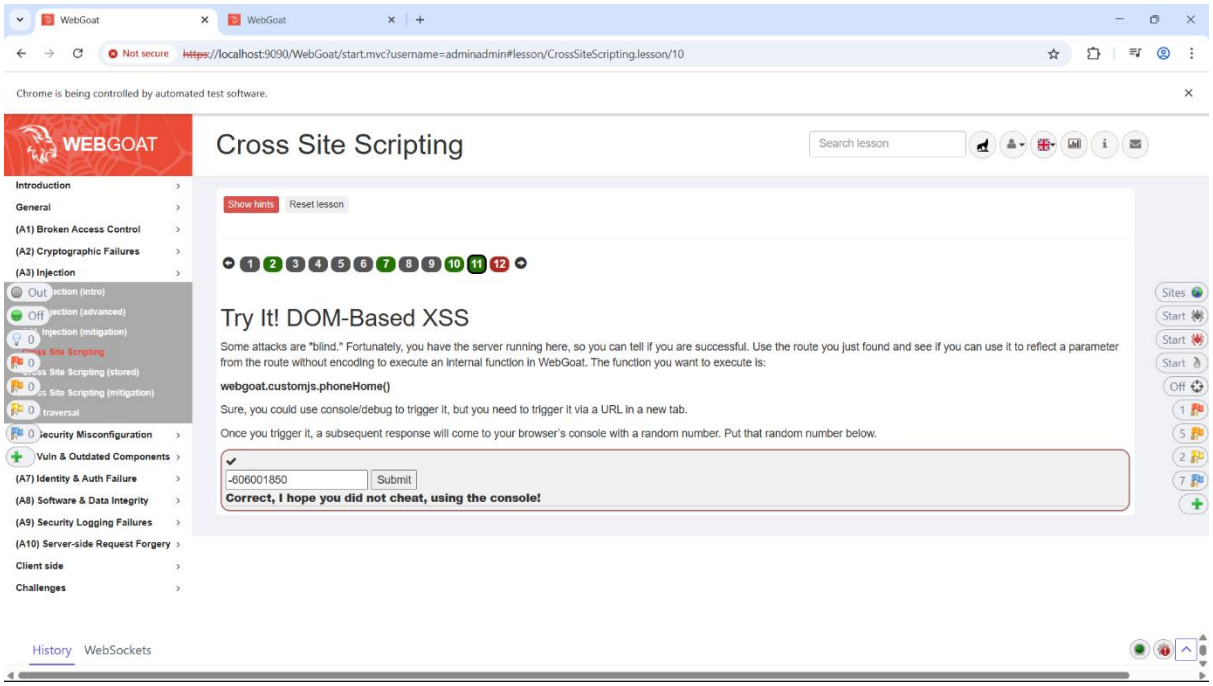
And this is the response query for that tested DOM attack.



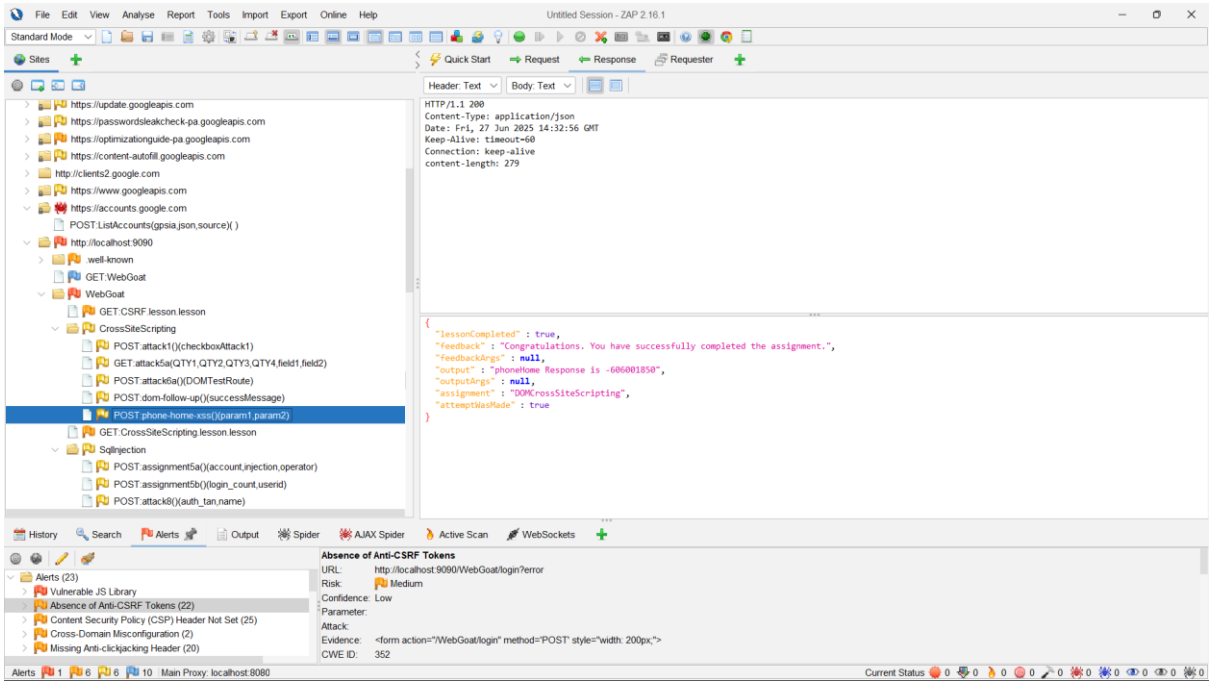
Now I open the exact path of start.mvc#test/ and I using the payload <script>webgoat.customjs.phoneHome()</script>.



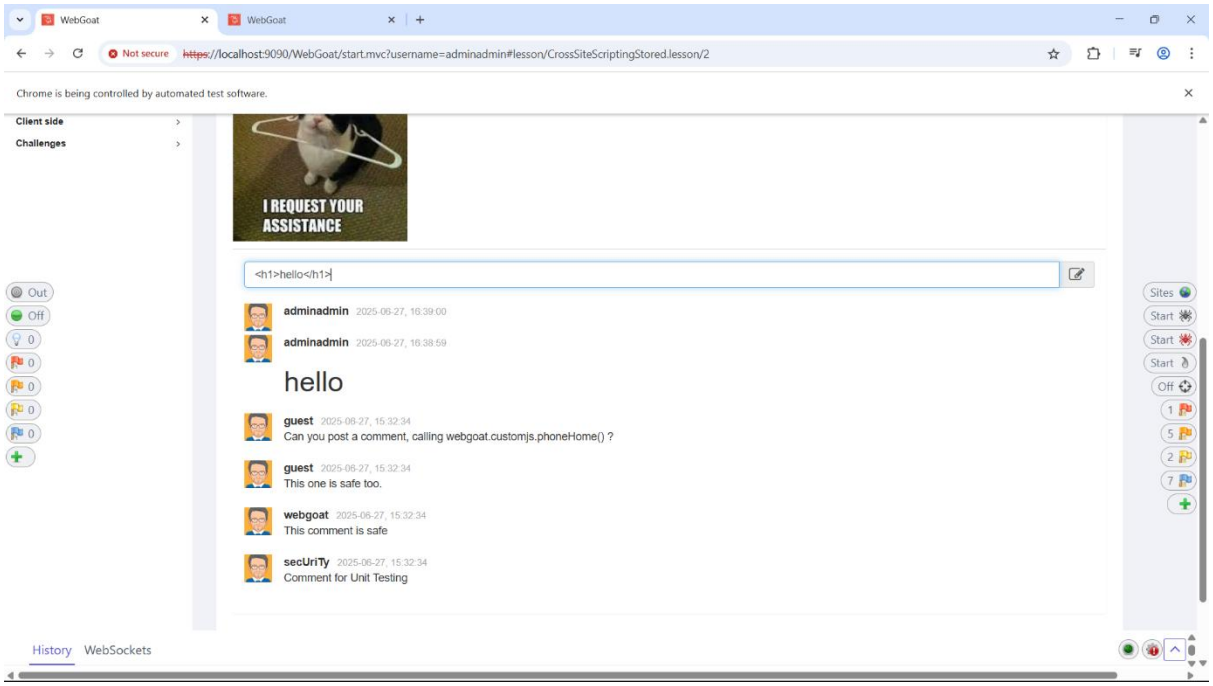
Successfully, it was executed and show the phoneHome = -60*.



Then I submit the PhoneHome and check that, finally it was correct and give that is true.



Stored-based XSS on http://localhost:9090/WebGoat/*/2 and the payload = <h1>Hello</h1>
It was executed successfully.



Mitigation:

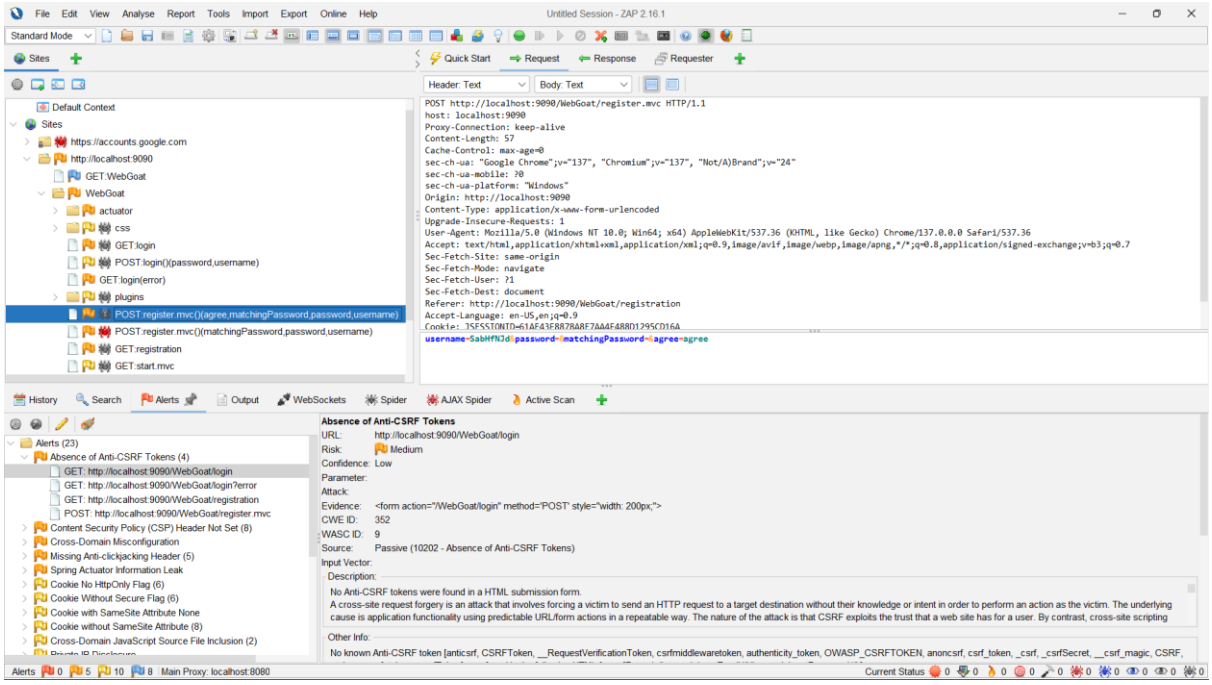
- Properly encode output to prevent script injection.
- Use Content Security Policy (CSP).
- Validate input and reject unsafe characters.

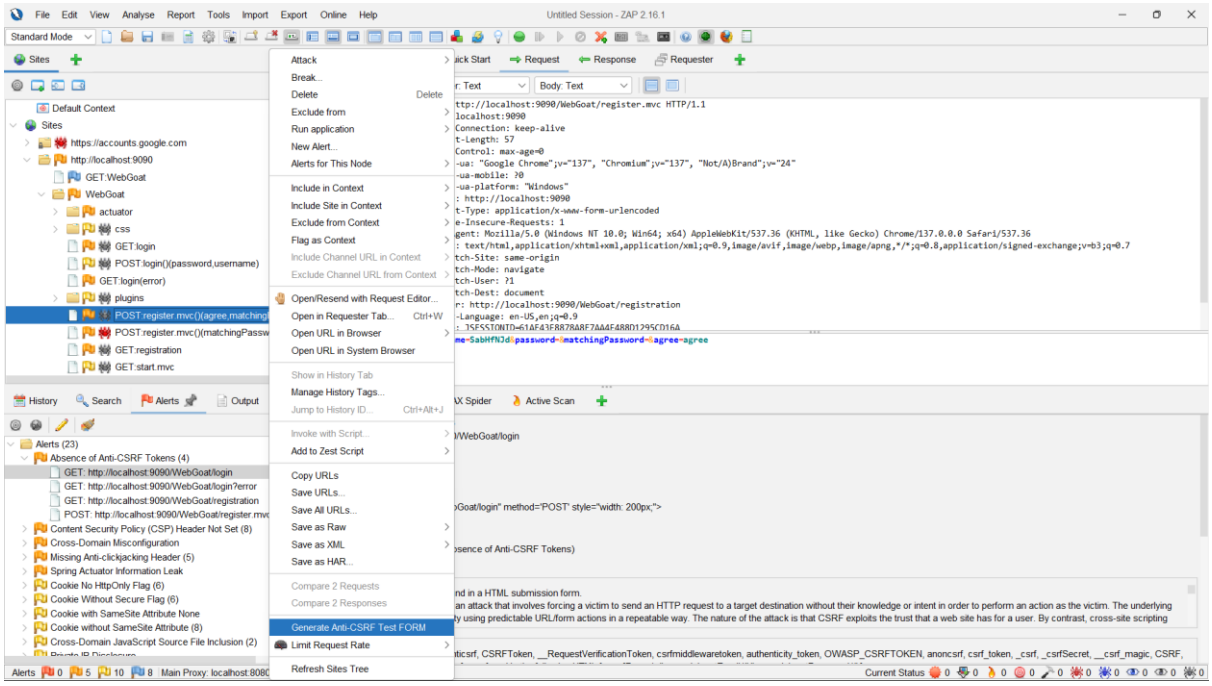
C. Cross-Site Request Forgery (CSRF)

- **Module:** WebGoat → CSRF
- Description: CSRF tricks a logged-in user into submitting a request unknowingly, which the server trusts.
- Test Used:
- <form action="http://localhost:8080/WebGoat/endpoint" method="POST">
- <input type="hidden" name="amount" value="10000">
- <input type="submit" value="Click me">
- </form>
- Observation:
  When opened in a logged-in session, the request succeeded without confirmation, proving CSRF.

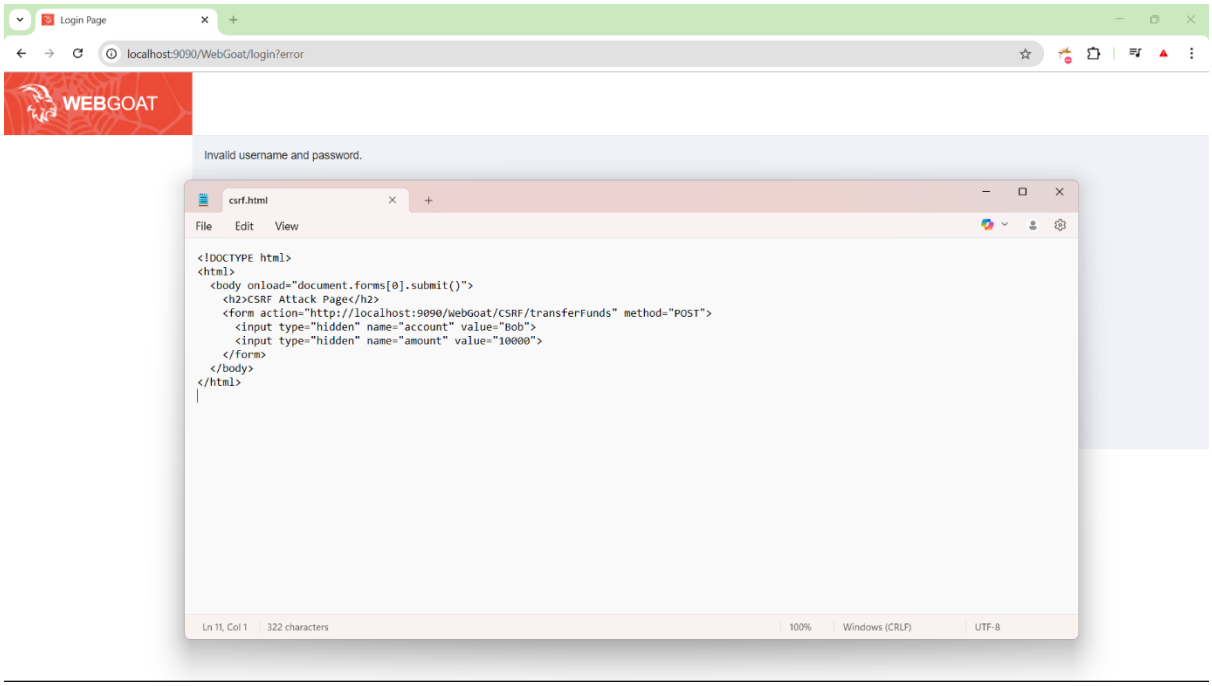
- Exploitation Steps:
    1. Captured the target request via ZAP.
    2. Replicated the request using a local HTML form.
- Impact:
  Can change user settings, perform financial transactions, etc.

Screenshot :

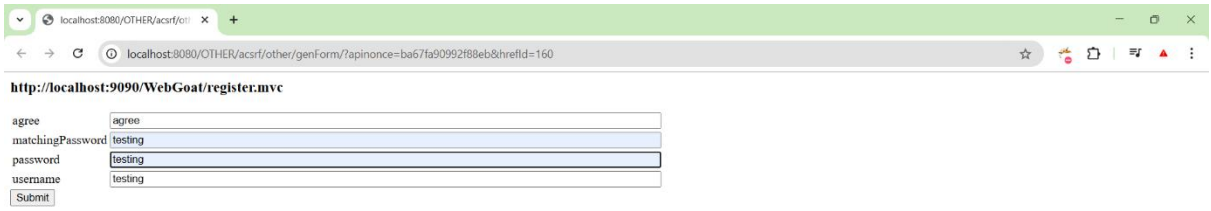


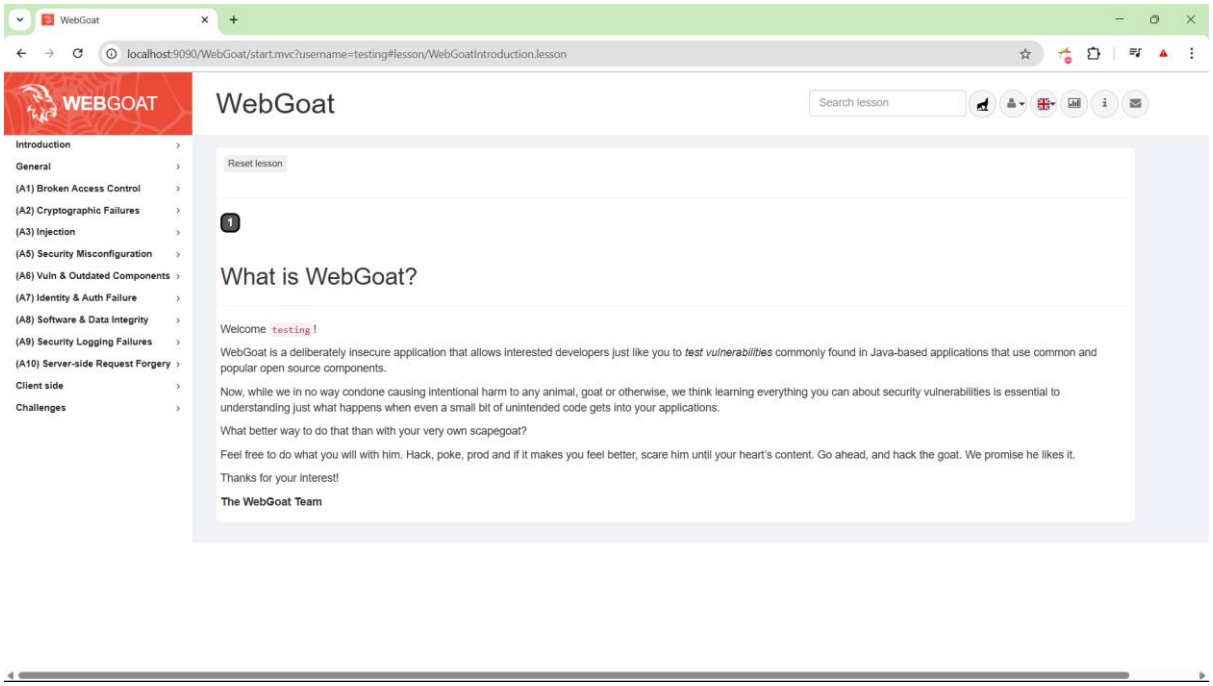


First capture the request on zap then make the poc for anti-csrf and open that is look like normal html form.



Then click the html file and open It you navigate the normal html page if you change any it must reflected on the active victim account pages.





Mitigation:

- Implement CSRF tokens (e.g., synchronizer tokens).
- Enforce SameSite cookie attribute.
- Validate the Referer or Origin headers.

4. Conclusion

Through this exercise, we identified and exploited three key vulnerabilities in WebGoat using OWASP ZAP and manual techniques. These vulnerabilities—SQL Injection, XSS, and CSRF—are among the OWASP Top 10 and highlight the importance of secure coding and input handling. Mitigating these issues requires both technical defenses (like parameterized queries and CSP headers) and a secure development lifecycle.

Task 3: LinkedIn Profile Engagement Report

Objective:

Completed Actions:

1. Followed Redynox on LinkedIn

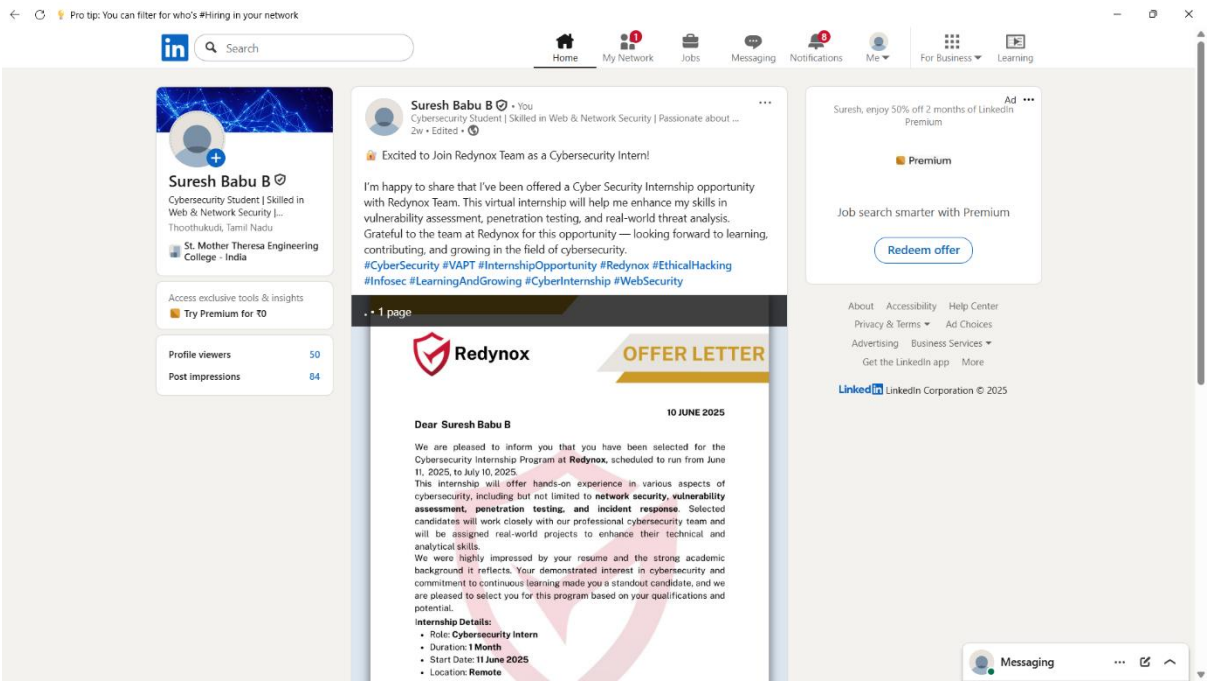
- Proof: Visible in the screenshot showing engagement with Redynox content.
- Outcome: Subscribed to Redynox's latest updates, strengthening brand connection.

2. Shared a Post About the Internship With a Personalized Message

Post Content:

🔐 Excited to Join Redynox Team as a Cybersecurity Intern!
I'm happy to share that I've been offered a Cyber Security Internship opportunity with Redynox Team. This virtual internship will help me enhance my skills in vulnerability assessment, penetration testing, and real-world threat analysis.
Grateful to the team at Redynox for this opportunity — looking forward to learning, contributing, and growing in the field of cybersecurity.

#CyberSecurity #VAPT #InternshipOpportunity #Redynox #EthicalHacking #Infosec #LearningAndGrowing #CyberInternship #WebSecurity



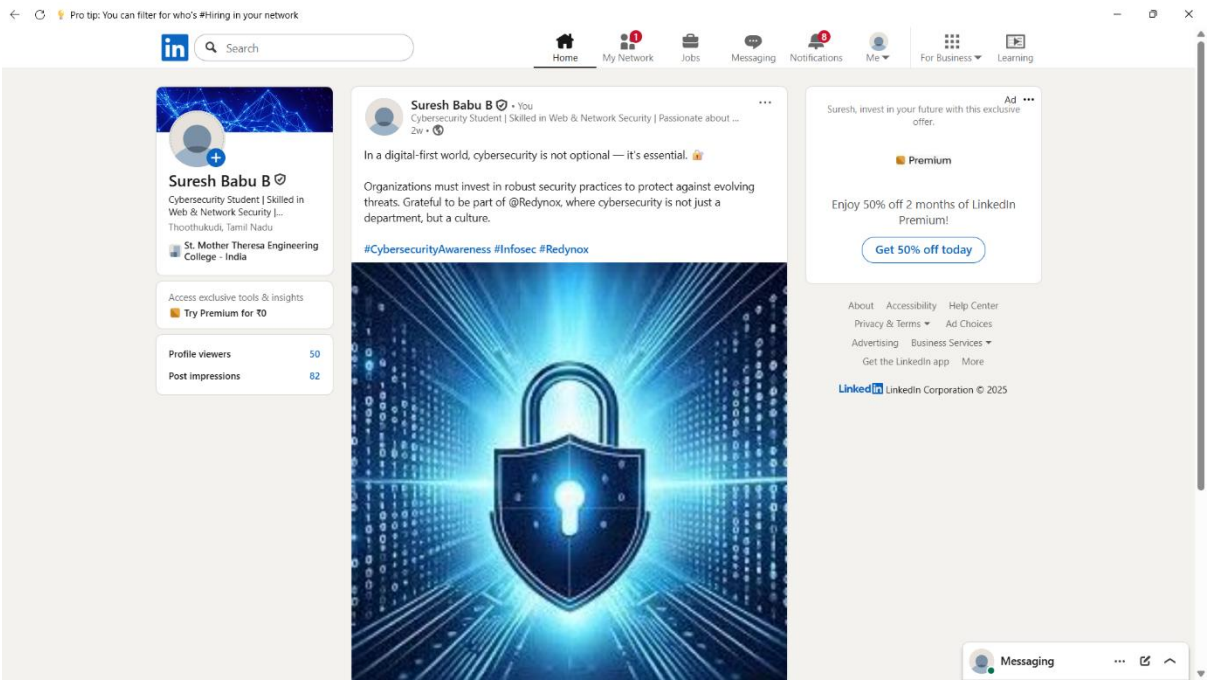
Impact: Displays professionalism, enthusiasm, and alignment with career goals.

3. Mentioned Redynox in a Post Discussing the Importance of Cybersecurity

Post Content:

In a digital-first world, cybersecurity is not optional — it's essential. 🔐
Organizations must invest in robust security practices to protect against evolving threats.
Grateful to be part of @Redynox, where cybersecurity is not just a department, but a culture.

#CybersecurityAwareness #Infosec #Redynox



Effect: Demonstrates commitment to spreading cybersecurity awareness.

Summary:

Task Item	Status	Evidence
Followed Redynox on LinkedIn	✅ Done	Screenshot (visible engagement)
Shared internship post with message	✅ Done	Screenshot (236).png
Mentioned Redynox in cybersecurity post	✅ Done	Screenshot (235).png & Full Post

Attached Proof: