****

**FUTURE INTERNS**

**CYBER SECURITY INTERNSHIP TASKS**

**(TRACK CODE: CS)**

**TASK 1 (WEB APPLICATION SECURITY TESTING)**

**Task: Conduct security testing on a sample web application to identify vulnerabilities like SQL injection, XSS, and authentication flaws.**

**Skills Gained: Web application security, ethical hacking, penetration testing.**

**Tools: SQLMap, kali Linux**

**Deliverable: A detailed security report with identified vulnerabilities and mitigation strategies.**

**I am going to conduct security testing on a sample vulnerable web application (**[**http://testfire.net**](http://testfire.net)**) and identify common web vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and Authentication Flaws, using tools like SQLMap and Kali Linux.**

**Step 1: Initial Reconnaissance using Nmap**

**We begin by scanning the target website or IP to gather information about open ports and running services.**

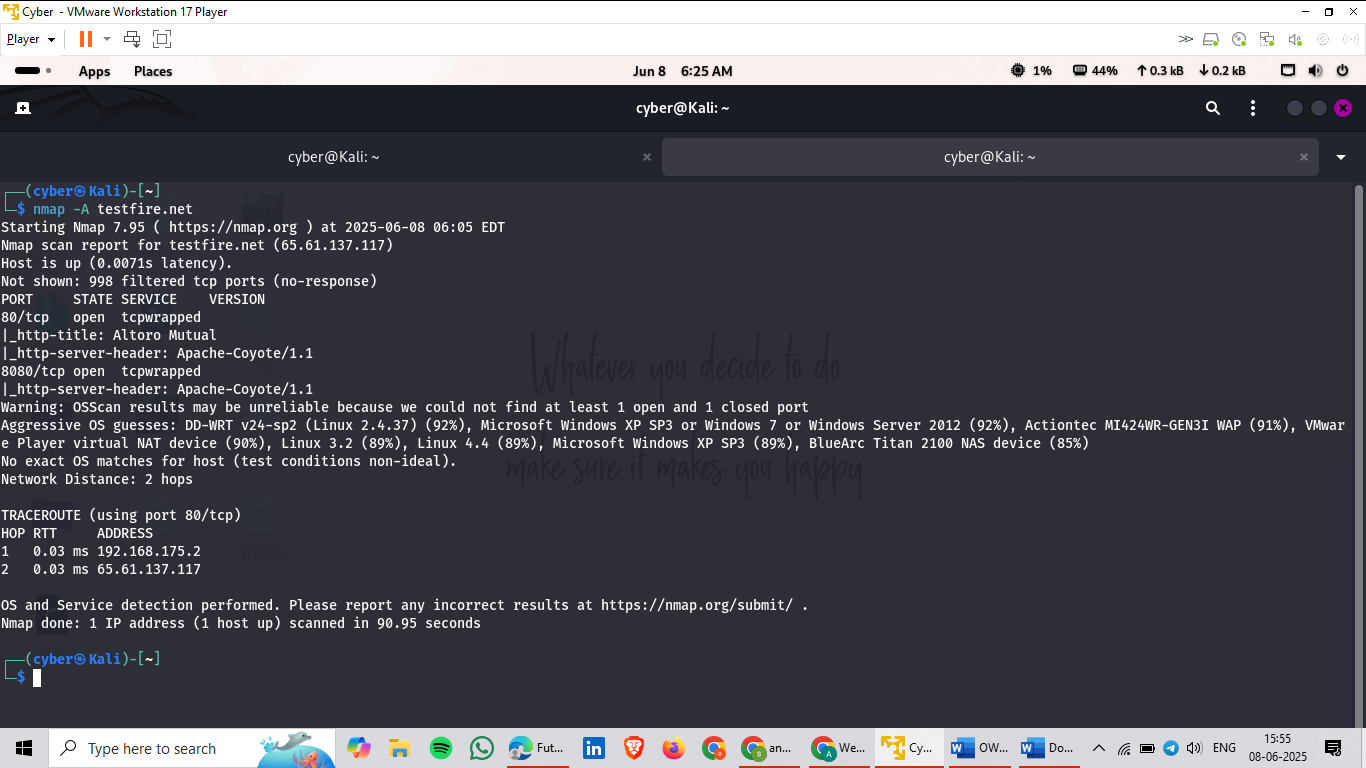
**✅ Command Used:**

**nmap -PN -sT -sV -p0-65534 65.61.137.117**

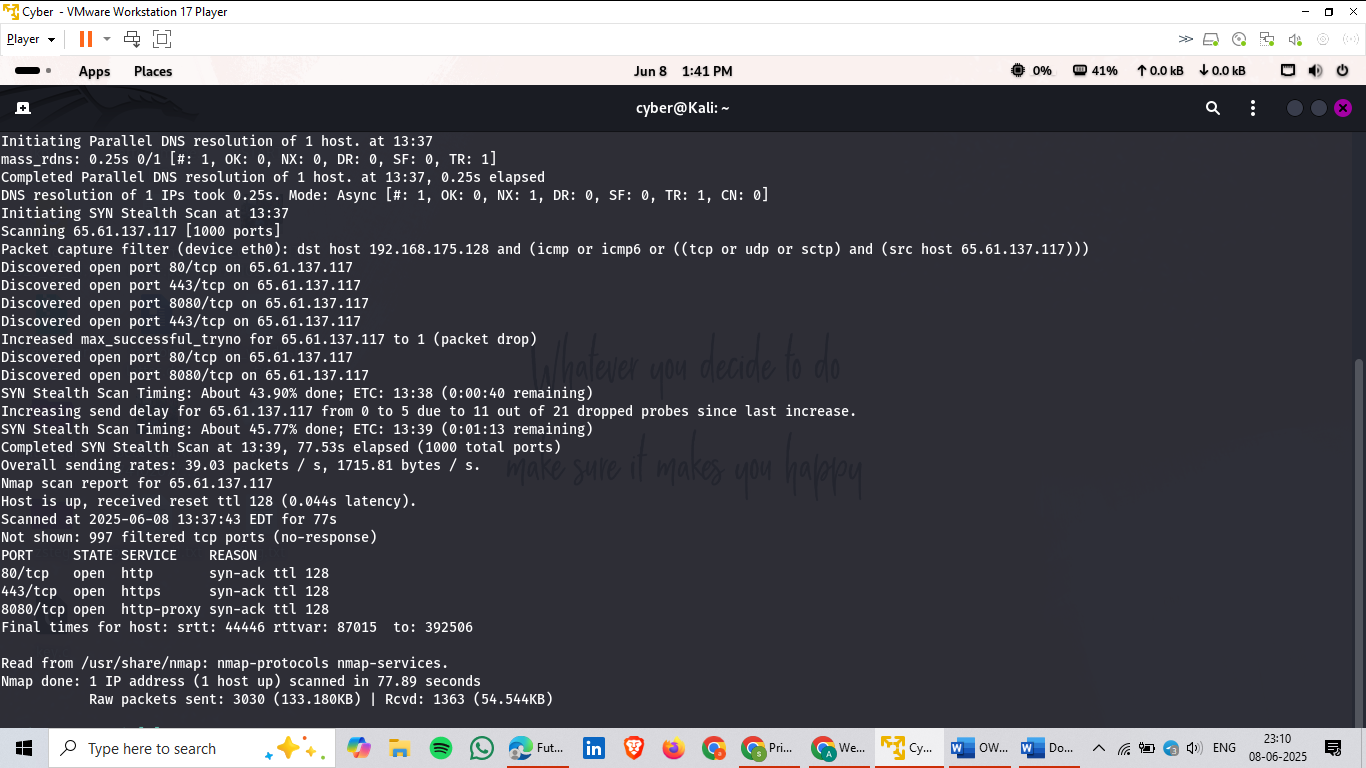
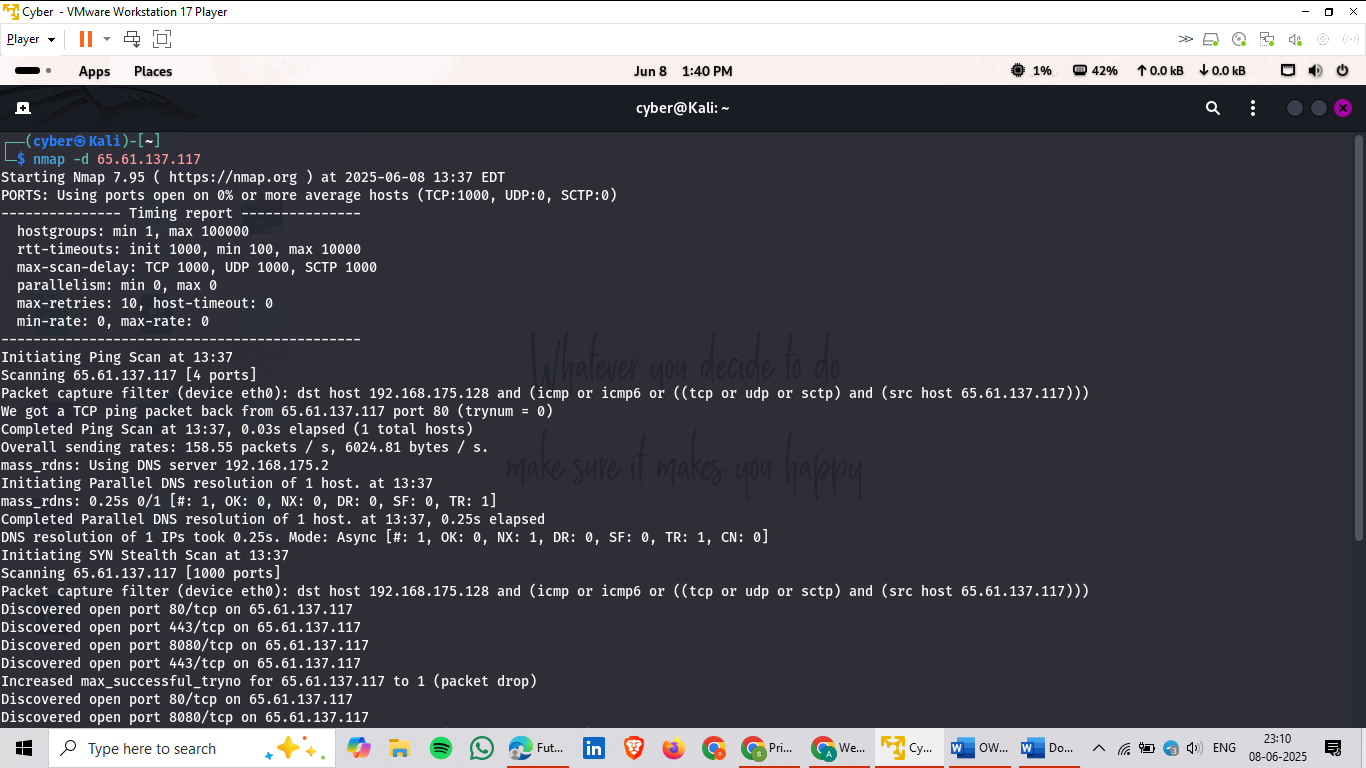
**What It Does:**

* **-PN: Skip host discovery (treat all hosts as online)**
* **-sT: TCP connect scan**
* **-sV: Version detection**
* **-p0-65534: Scan all 65535 ports**

**🔎 Purpose: To enumerate all open ports and identify services running on them, which may be vulnerable.**

****

**Then we run a decoy command for this particular ip..**

****

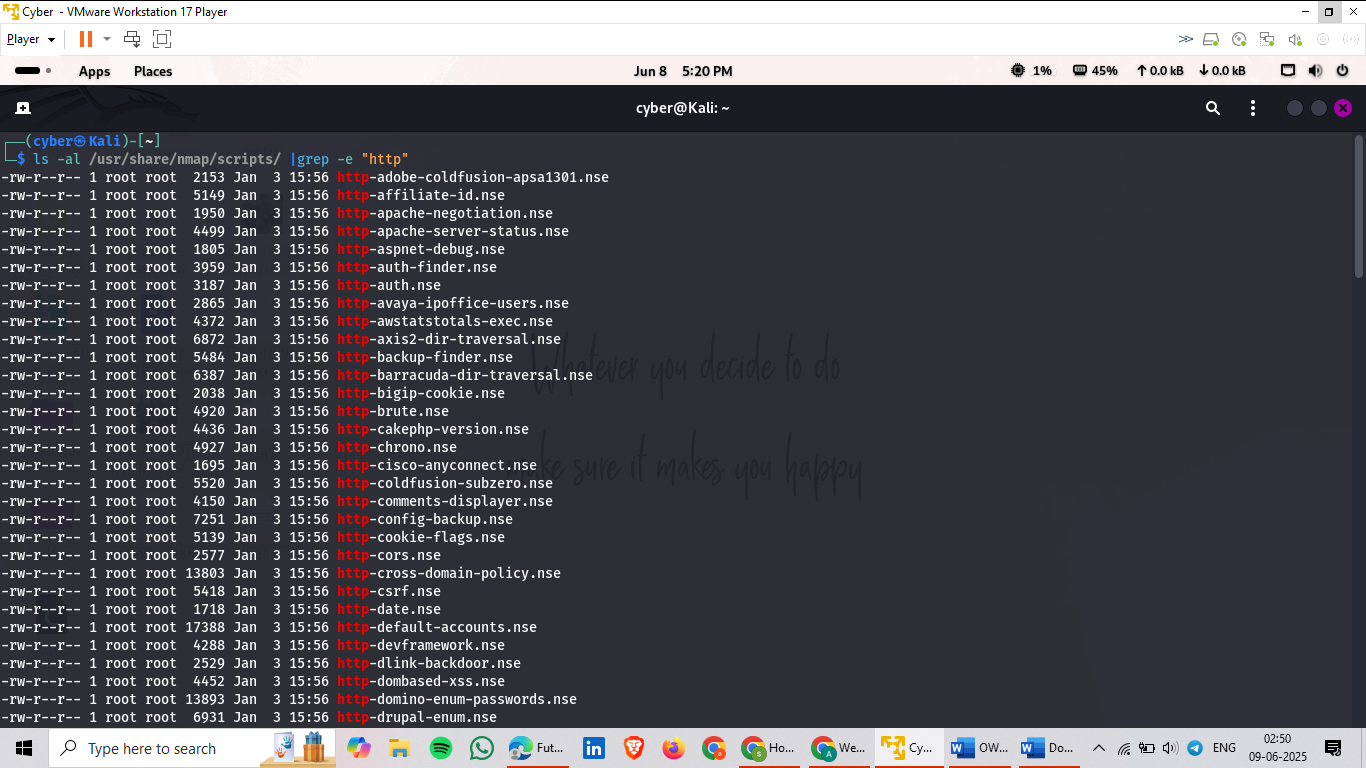
**We use Nmap script command to check potential script of http to run and find more vulnerabilities.**

**ls -al /usr/share/nmap/scripts**

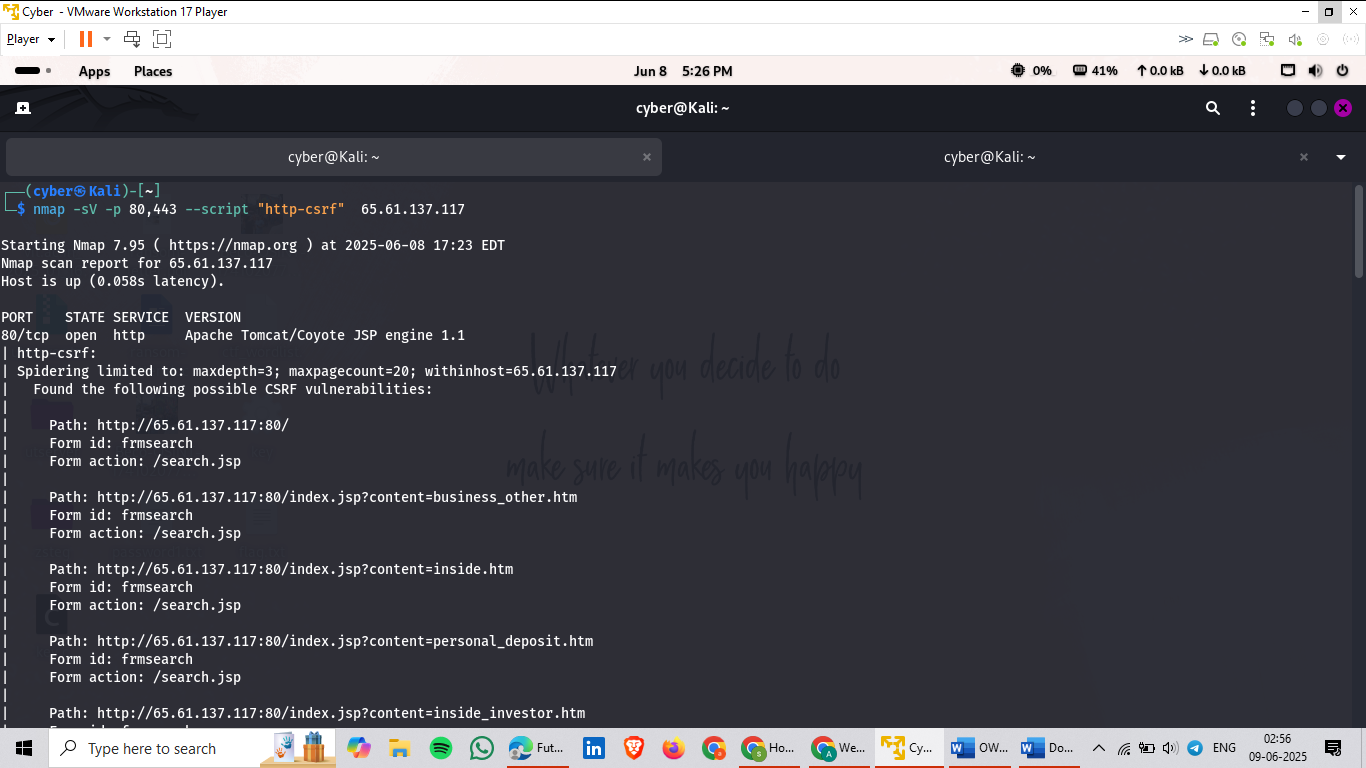
**ls -al /usr/share/nmap/scripts |grep -e http**

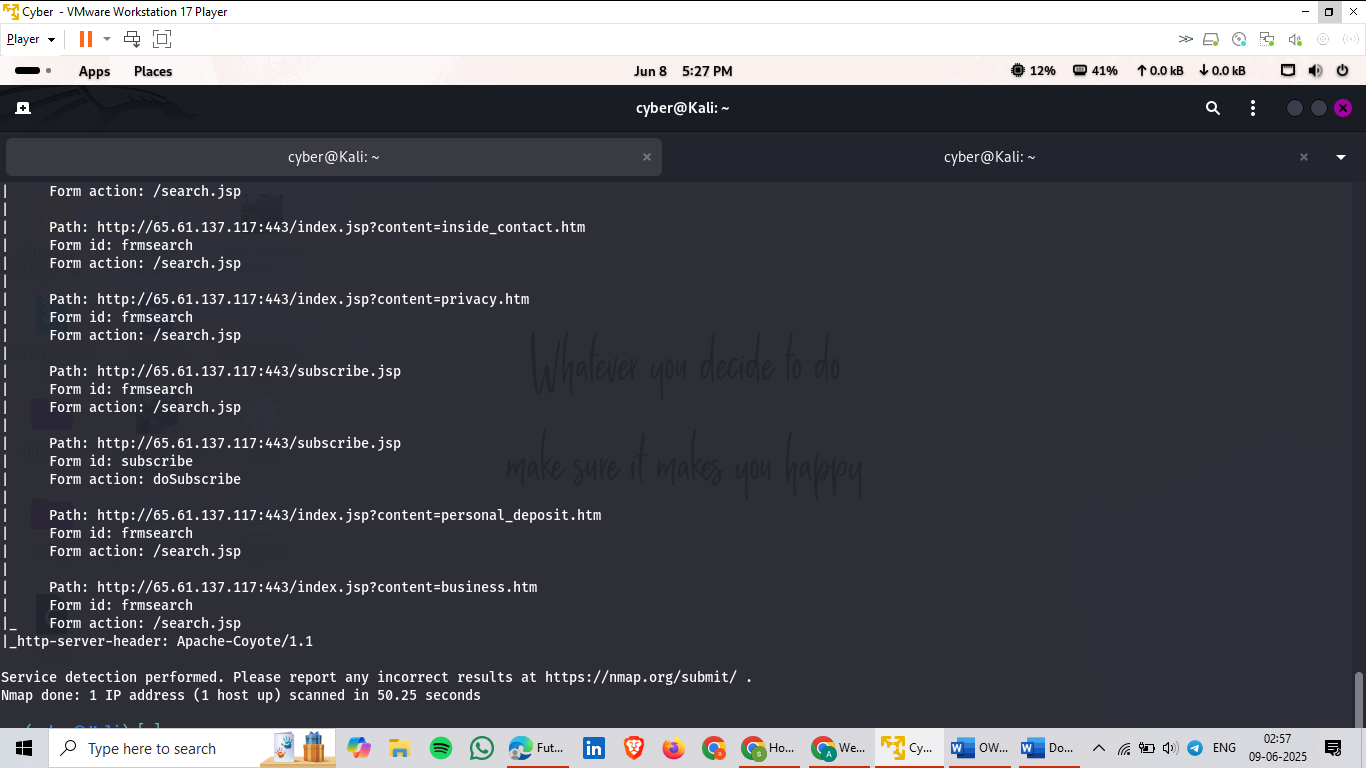
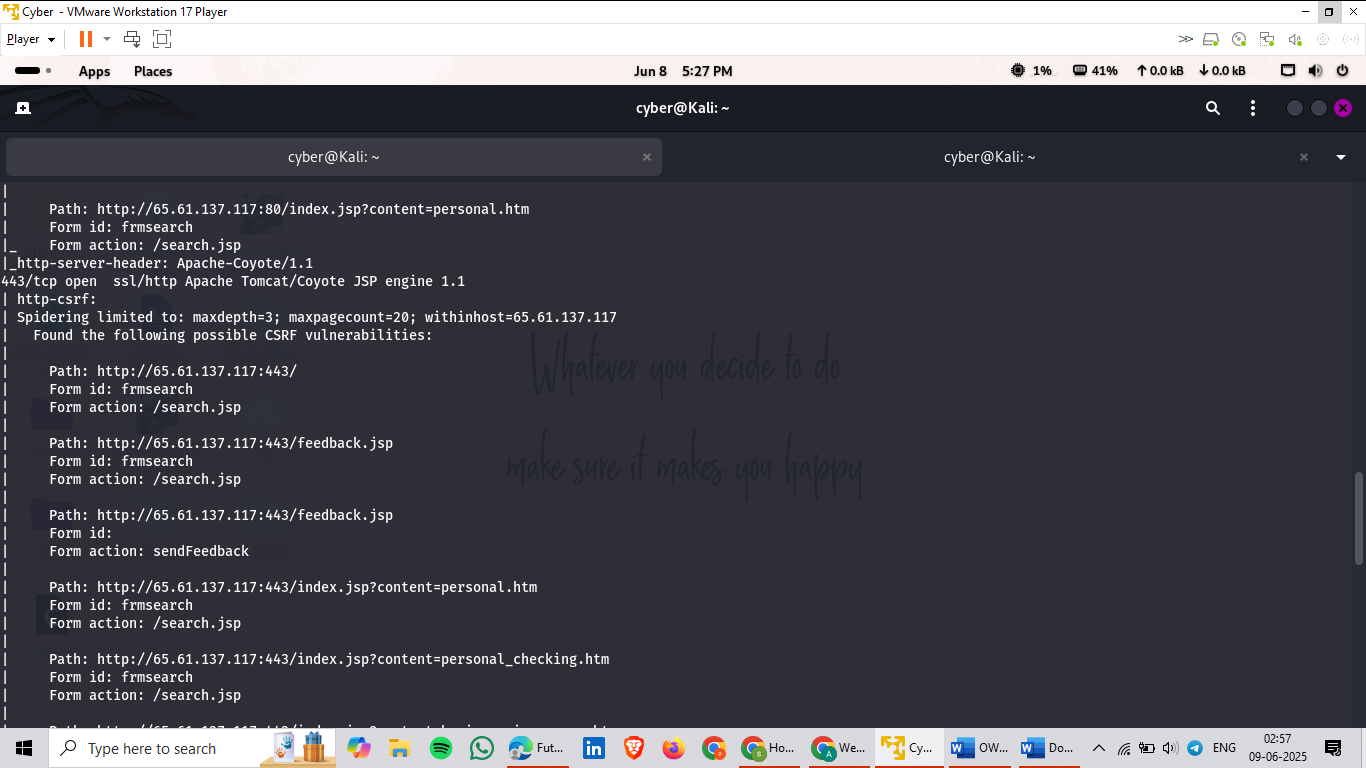
**Then we need to use this command**

**nmap -sV -p 80,443 --script "http-sql-injection,http-csrf" <target-ip>**

****

**We now find a csrf vulnerabilities**

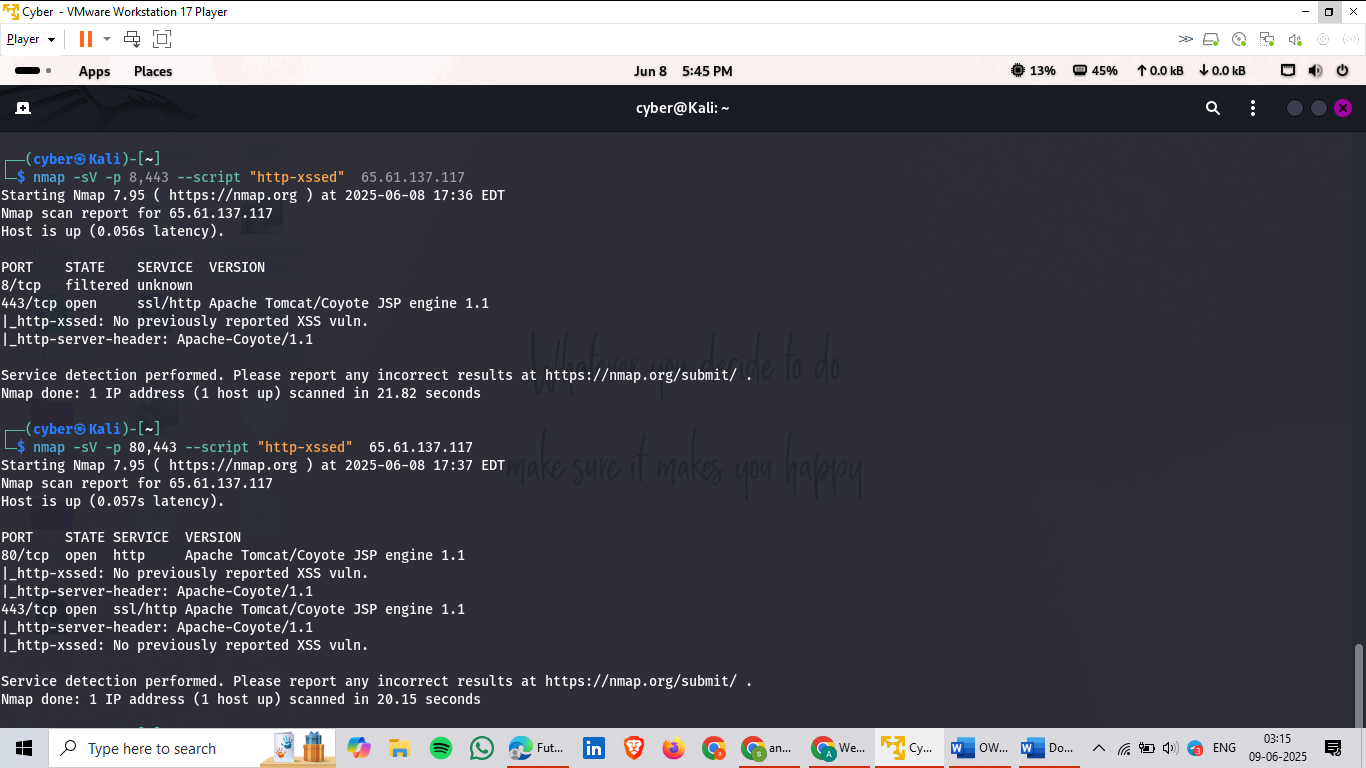
****

****

**Now we trying to find whether this website ever report for xss vulnerabilities or not using this command.**

**nmap -sV -p 8,443 --script "http-xssed" 65.61.137.117**

**This will show there is no vul.regarding xss**

****

**Now go for other vulnerability.**

**Step 2: Vulnerability Scanning with OWASP ZAP**

**After the Nmap scan, we use OWASP ZAP to perform a deeper scan of the web application.**

**🛠 How We Do It:**

1. **Open OWASP ZAP.**
2. **Enter the target URL (e.g., http://testfire.net) into the URL box.**
3. **ZAP then crawls the website to understand its structure.**

**🔎 What It Finds:**

* **Pages, directories, files**
* **Login forms, change-password pages**
* **Functionalities where users interact (attack surface)**

**🕸️ Step 3: Spidering the Application**

**We enable the Spider feature in OWASP ZAP.**

**🧾 Purpose:**

* **To extract all internal URLs from the website.**
* **Discover all possible entry points for vulnerability scanning.**

**⚙️ Step 4: Active Scanning**

**Once URLs are gathered using Spider:**

* **We send them to Active Scan.**
* **ZAP scans each URL for vulnerabilities automatically.**

**🧪 Mode Used:**

* **Standard Mode (sufficient for intentionally vulnerable apps like testfire.net)**
* **For complex or real-world applications, advanced modes may be used.**

**🔐 Step 5: Understanding Vulnerability Severity in ZAP**

**OWASP ZAP color-codes the vulnerabilities:**

| **Color** | **Meaning** |
| --- | --- |
| **🔵 Blue** | **Informational leaks (e.g., headers, banners, comments)** |
| **🟡 Yellow** | **Low severity vulnerabilities (non-critical but still need fixing)** |
| **🟠 Orange** | **Medium severity (can be exploited; needs immediate attention)** |
| **🔴 Red** | **High severity (critical issues such as SQLi, XSS, Auth flaws)** |

**🧠 Key Insight:**

**"The more functionality a website has, the higher the chance of vulnerabilities. More features = more attack surface."**

**📁 Step 6: Exporting the Results**

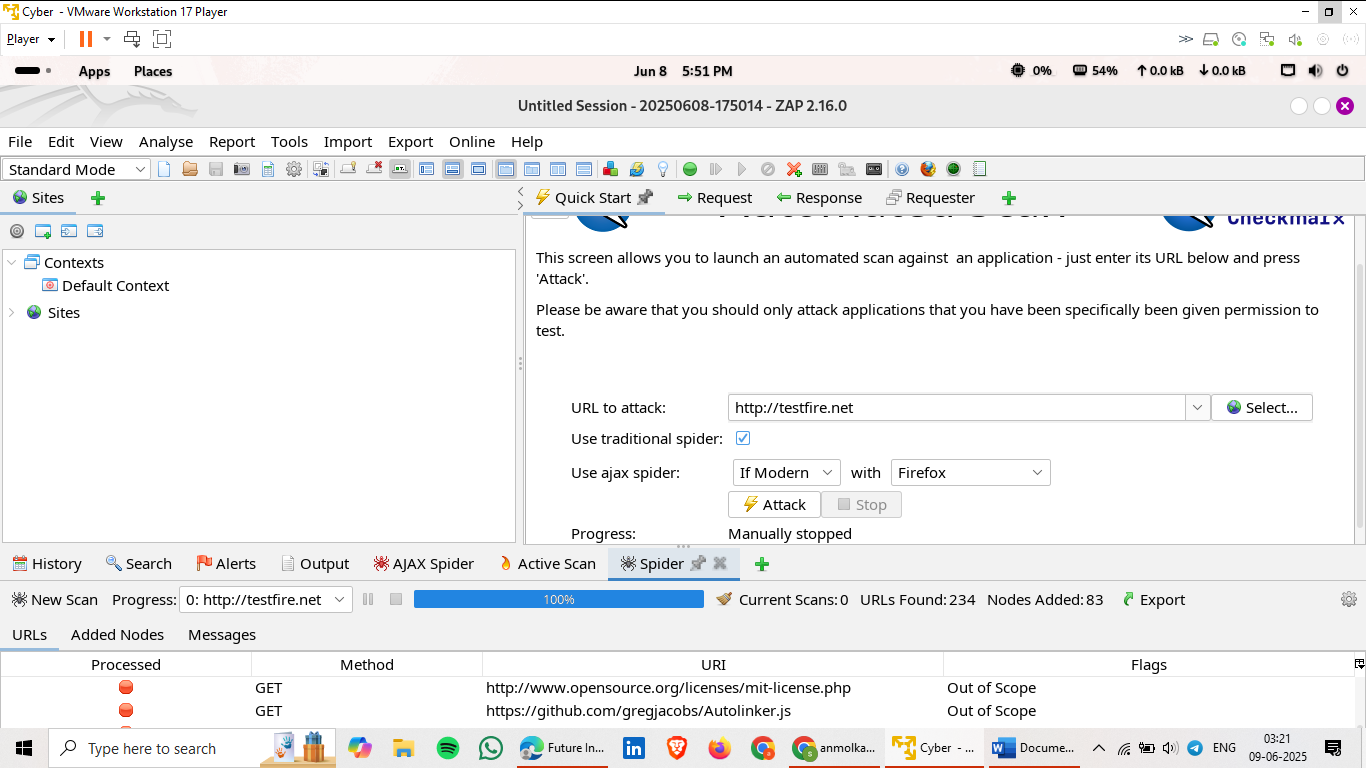
**After scanning:**

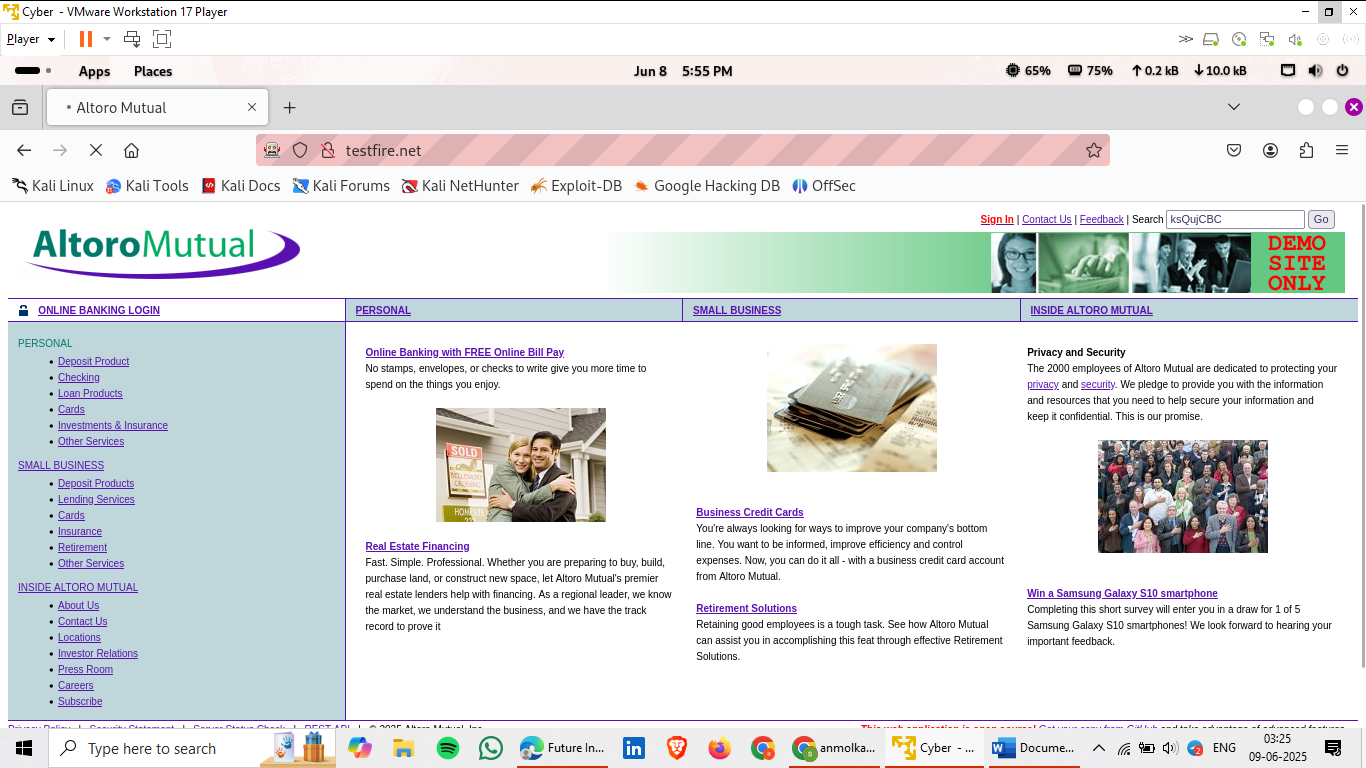
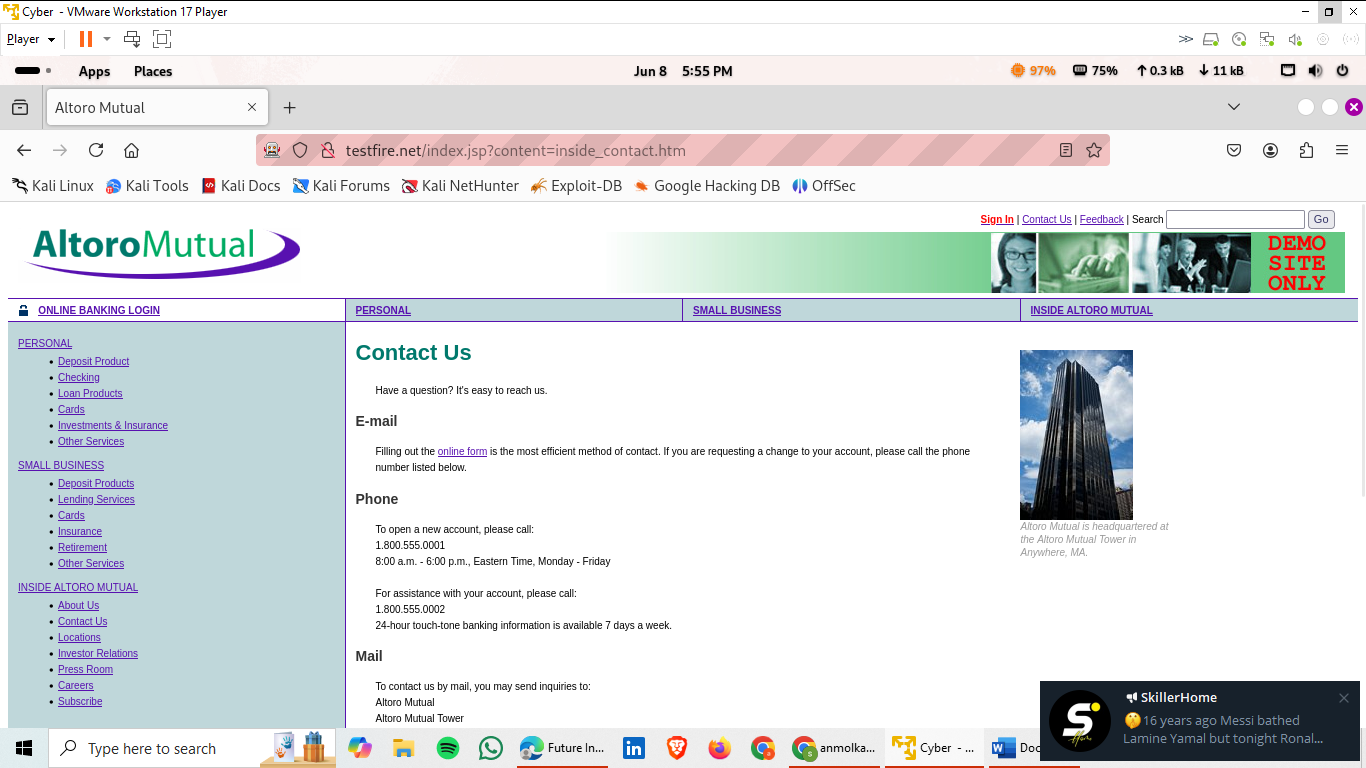
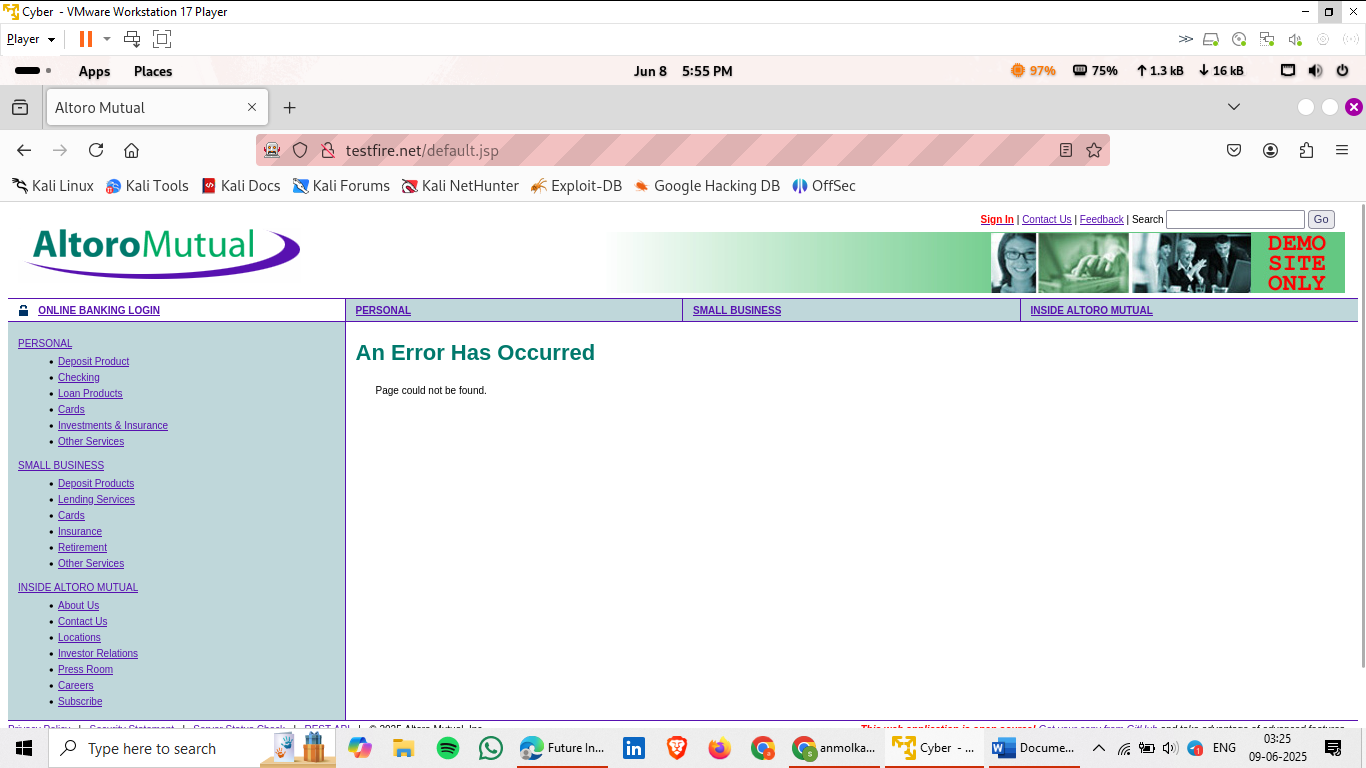
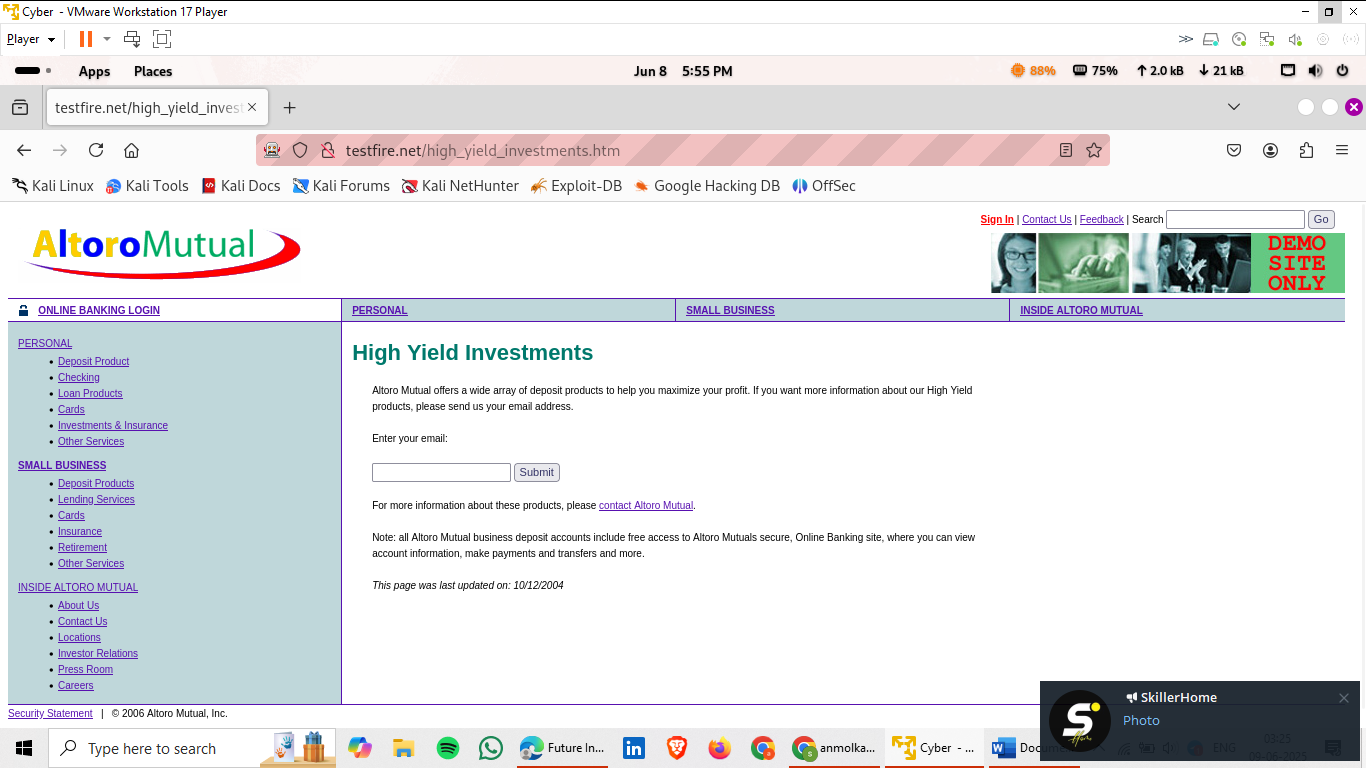
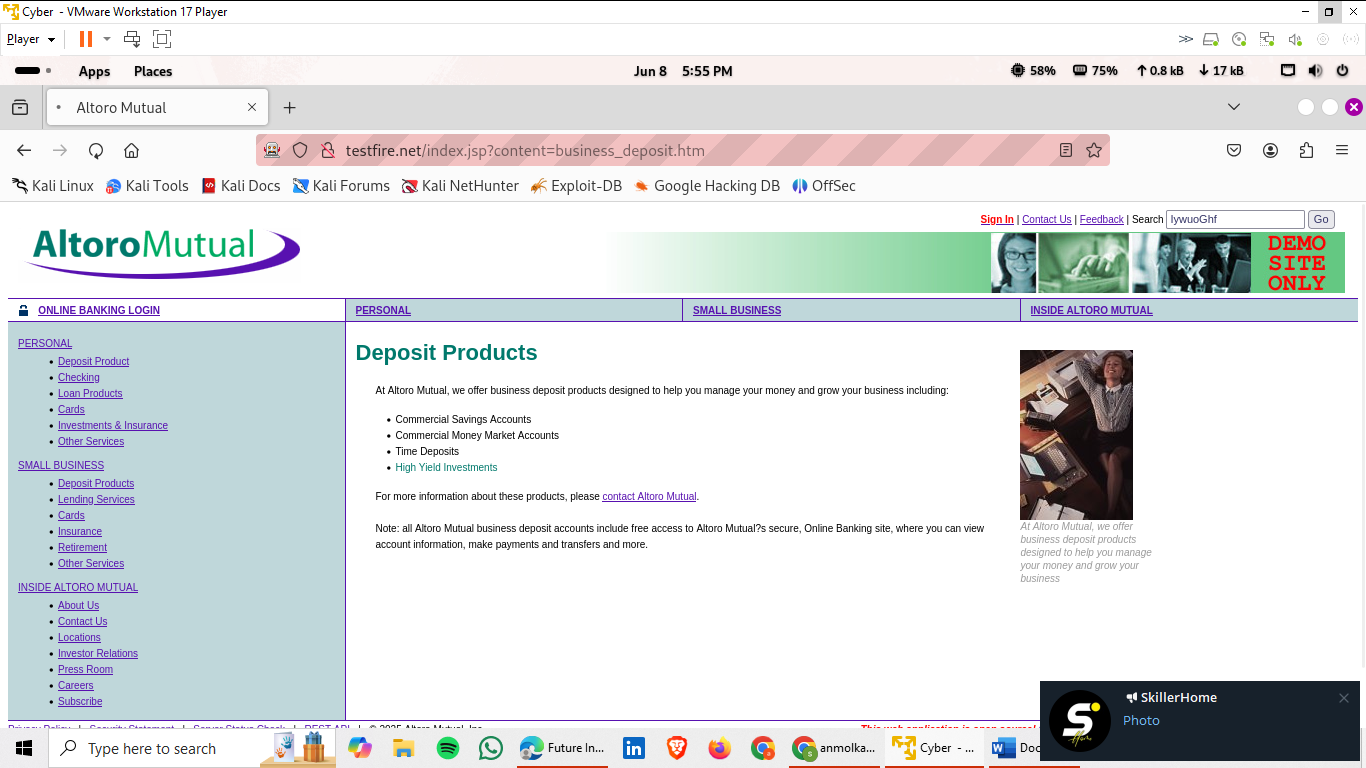
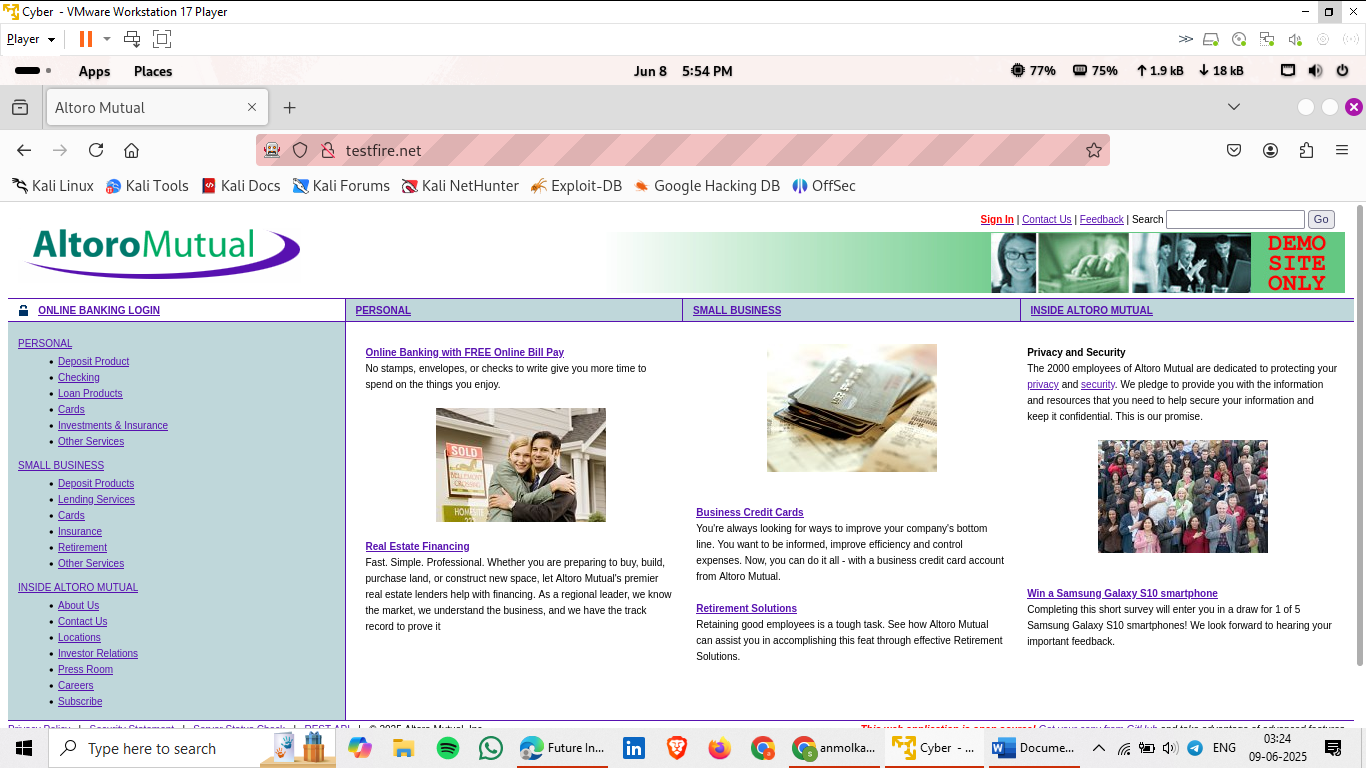
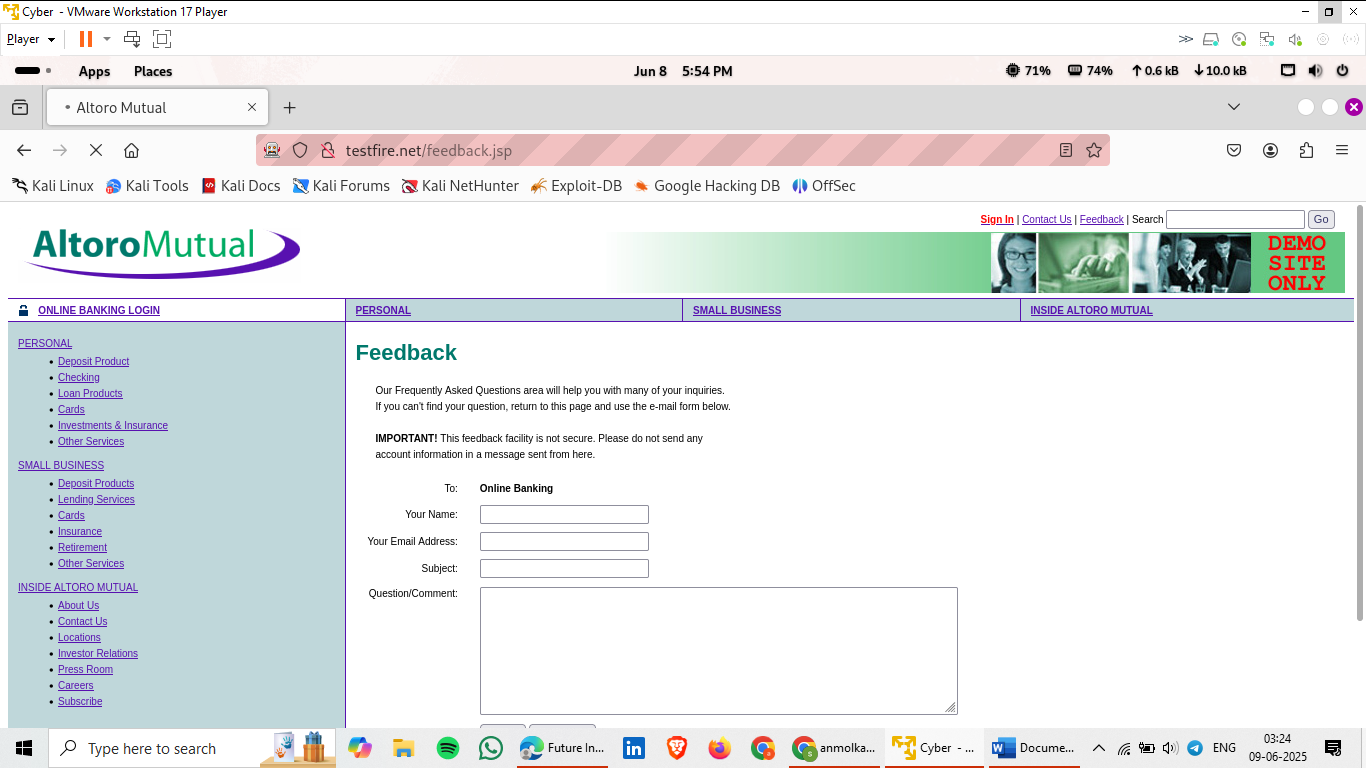
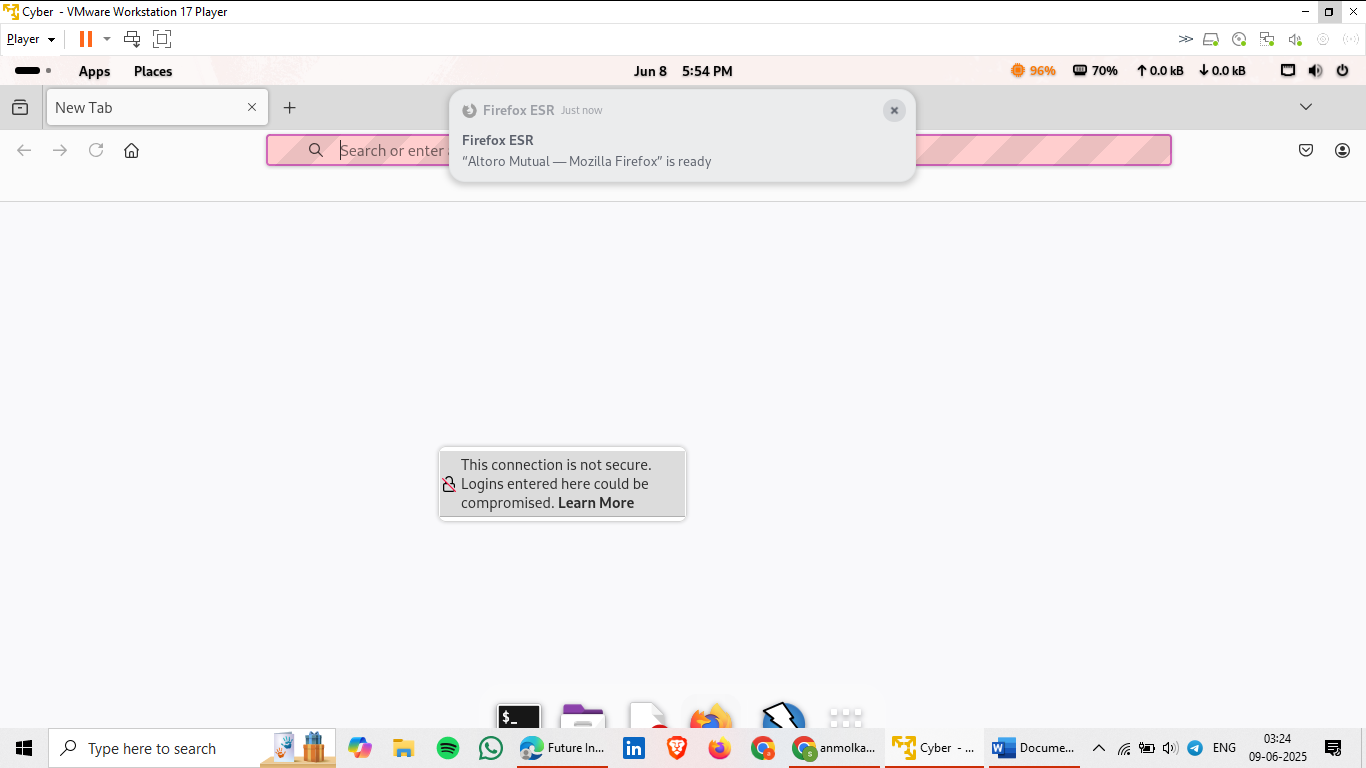
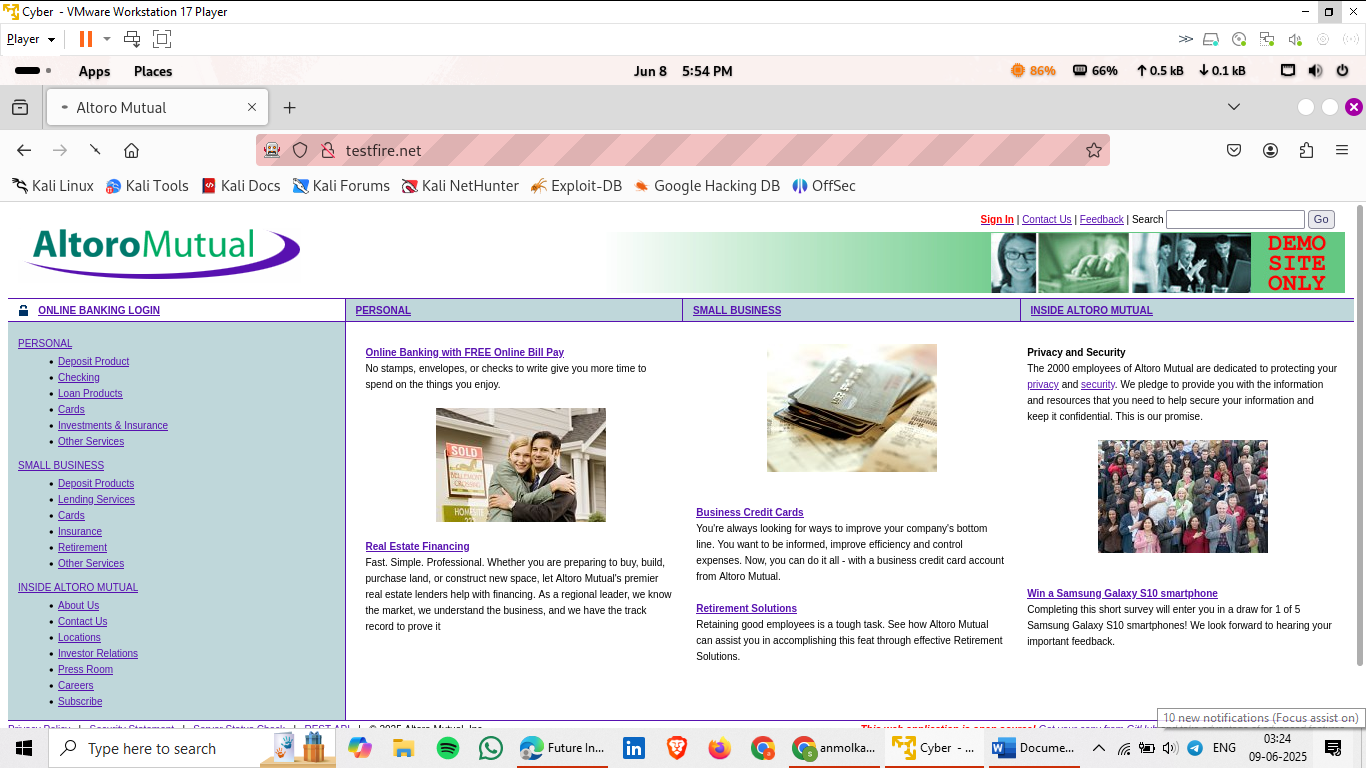
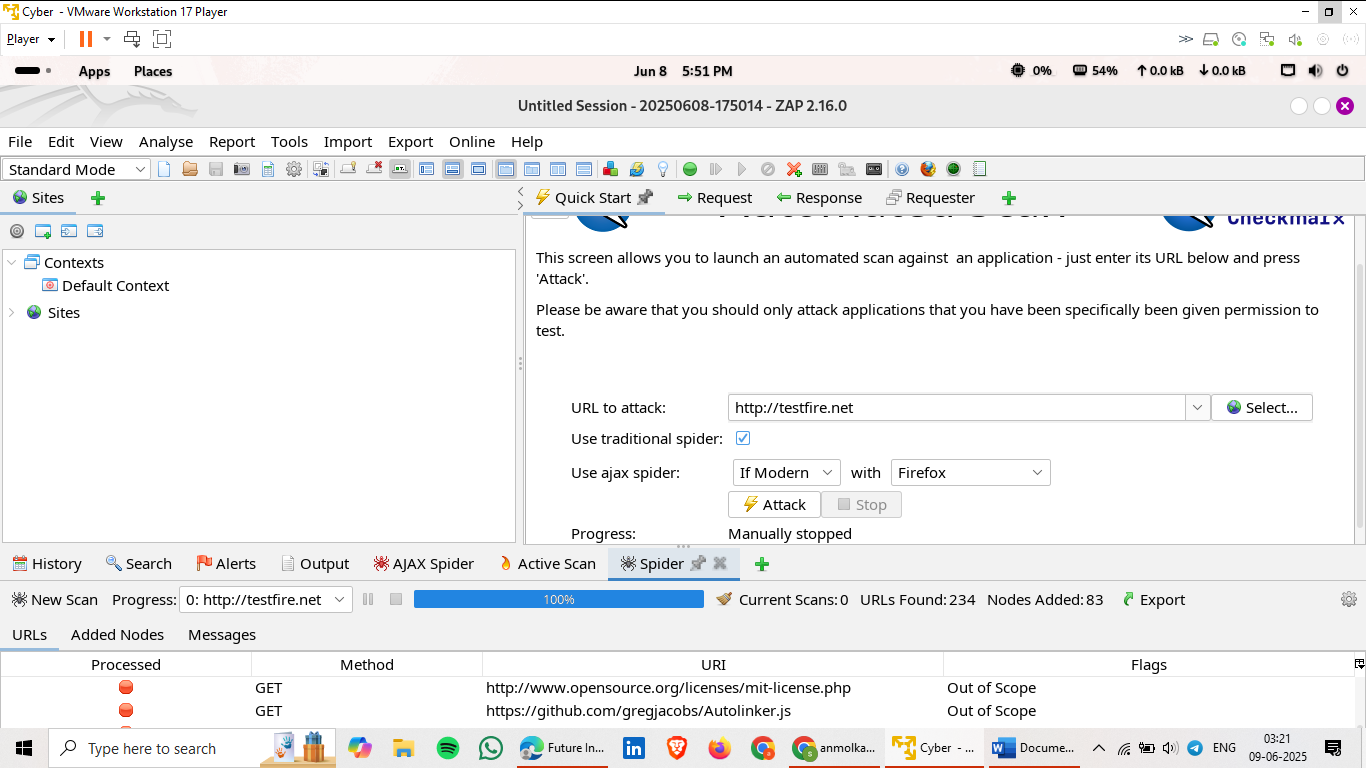
* **Export the report in CSV or Word format.**
* **This file includes all found vulnerabilities, their details, severity, and suggested fixes.**

**🛡️ Step 7: Reviewing and Applying Mitigation Techniques**

**For each vulnerability found:**

* **Analyze the risk**
* **Suggest mitigation strategies like:**
  + **Input validation**
  + **Proper session management**
  + **Strong authentication**
  + **Secure headers**
  + **Disabling directory listing**

****

****

**This was my final submission. I have tried my best to complete the task as it is my first time doing internship work. If anything is not up to the mark, please feel free to inform me via email or any other channel—I will work on improving it. I believe that learning from feedback is the most valuable part of the journey.**