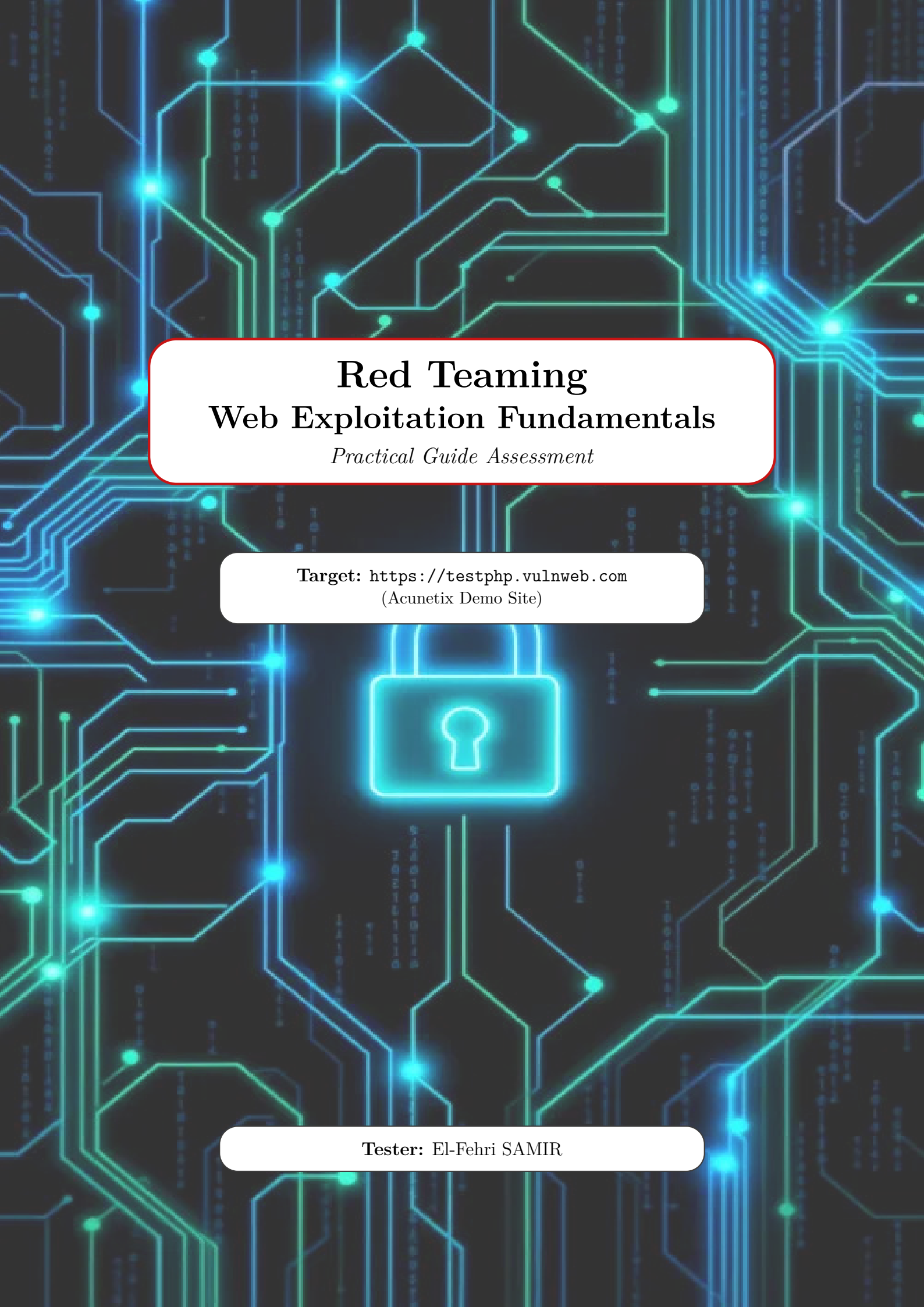

Red Teaming
Web Exploitation Fundamentals

Practical Guide Assessment

Target: `https://testphp.vulnweb.com`
(Acunetix Demo Site)

Tester: El-Fehri SAMIR

Contents

List of Figures

1 Introduction

The website testphp.vulnweb.com is a deliberately vulnerable web application hosted by Acunetix for educational and testing purposes. It simulates common web vulnerabilities such as SQL injection, cross-site scripting (XSS), and weak authentication to help security professionals and learners practice penetration testing. The site is built with PHP and MySQL, and users can explore vulnerabilities by interacting with features like login forms, search fields, and product listings. It serves as a safe environment to test tools like SQLmap, Burp Suite, and OWASP ZAP.

So our mission in this task is trying to find , enumerate and exploit these vulnerabilities using common pentesting tools on kali linux environment , to get understand how these weaknesses working mechanisms for each one .
In next sections we follow a process typically involves a series of well-defined phases, with variations in the number and naming of steps depending on the methodology used , like this steps strategy :

1. Reconnaissance
2. Scanning and Enumeration
3. Vulnerability Assessment
4. Vulnerability Assessment

So keep reading the documentation report to discover our results, and progress in penetration testing skills.

2 Reconnaissance

In this phase, we gather as much information as possible about the target environment from publicly available sources. This includes using techniques like OSINT (Open Source Intelligence), searching public records, analyzing domain information, reviewing social media, and using tools like WHOIS lookups and Google dorking to map potential attack surfaces.

2.1 Burp Suite

BurpSuite is one of the powerful web reconnaissance and information gathering tools , gives the tester huge options to get information with integrated browser. So we use it as the first tool to get used software versions and details as showing in the screenshot:

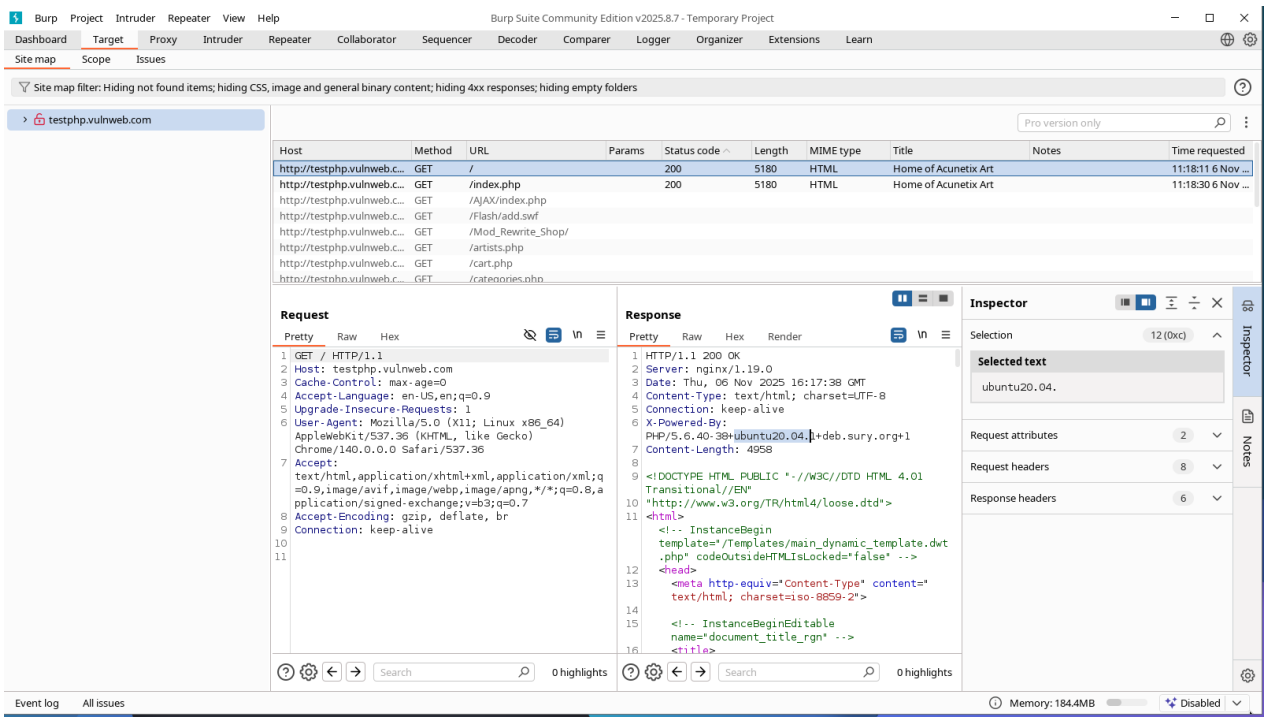

Figure 2.1: BurpSuite reconaissance results

When we send a request to target we get an HTTP response with technologies receipt used in this website , as showed in the picture we have :

- Web server : `Nginx` version `1.19.0`
- Programming language : `PHP` version `5.6.40` (vulnerable)
- Operating system : `Ubuntu` version `20.04`

3 Scanning and Enumeration

Following reconnaissance, we need to use automated tools to actively probe the target systems. This step involves network scanning to identify open ports and services, and enumeration to gather detailed information about the identified services.

3.1 Nmap scanning

Nmap gives us a super technique guide to scan network and host open ports and services, and this is the result of our nmap scan :

```
┌──(kali㉿kali)-[~]
└─$ nmap -sS -sC -sV  testphp.vulnweb.com
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-06 06:23 EST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.023s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
Not shown: 999 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.19.0
|_http-title: Home of Acunetix Art

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 45.52 seconds
```

Figure 3.1: Nmap scanning results

3.2 Gobuster

Using Gobuster we can scan and discover host directories and files structure , and this is the results after scan the target :

3.3 Nikto

Also in addition to nmap and gobuster there is the result of Nikto :

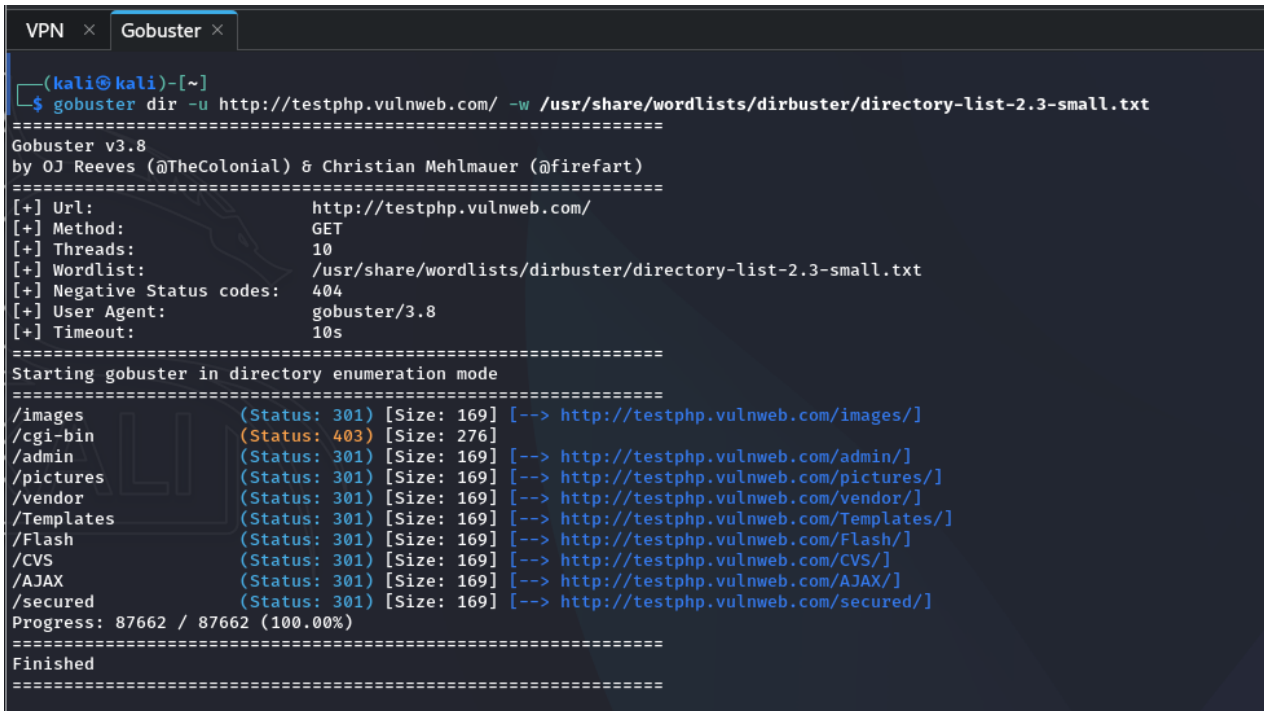

Figure 3.2: Gobuster scan results

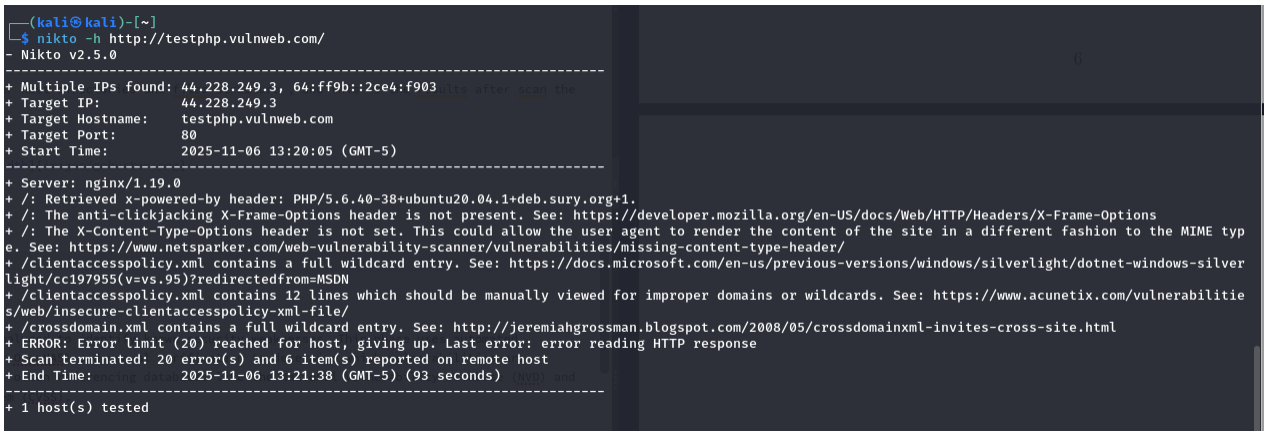

Figure 3.3: Nikto results

4 Vulnerability Assessment

The data collected from scanning is analyzed to identify known security weaknesses. This phase uses automated vulnerability scanners like Nessus or OWASP ZAP and manual verification to determine which vulnerabilities are exploitable and pose the highest risk. After some searching we get some common vulnerabilities for finded software versions like :

Software	**Version**	**CVE code**	**vulnerability type**
Nginx	1.19.0	CVE-2021-23017	Denial of Service (DoS)
PHP	5.6.40	CVE-2018-15133, CVE-2017-16894, CVE-2013-7345	RCE, Integer/Buffer overflow, Privilege Escalation
Ubuntu	20.04	CVE-2020-25712	Privilege Escalation

For other common web attacks like XSS, SQLi, and CSRF we decide to use OWASP ZAP tool with automated attack. With automated attack technique we discover other attacks with score level of each one as the highest one is XSS, Medium level there is CSRF and clickjacking , in addition to the complete list as showing in next image:

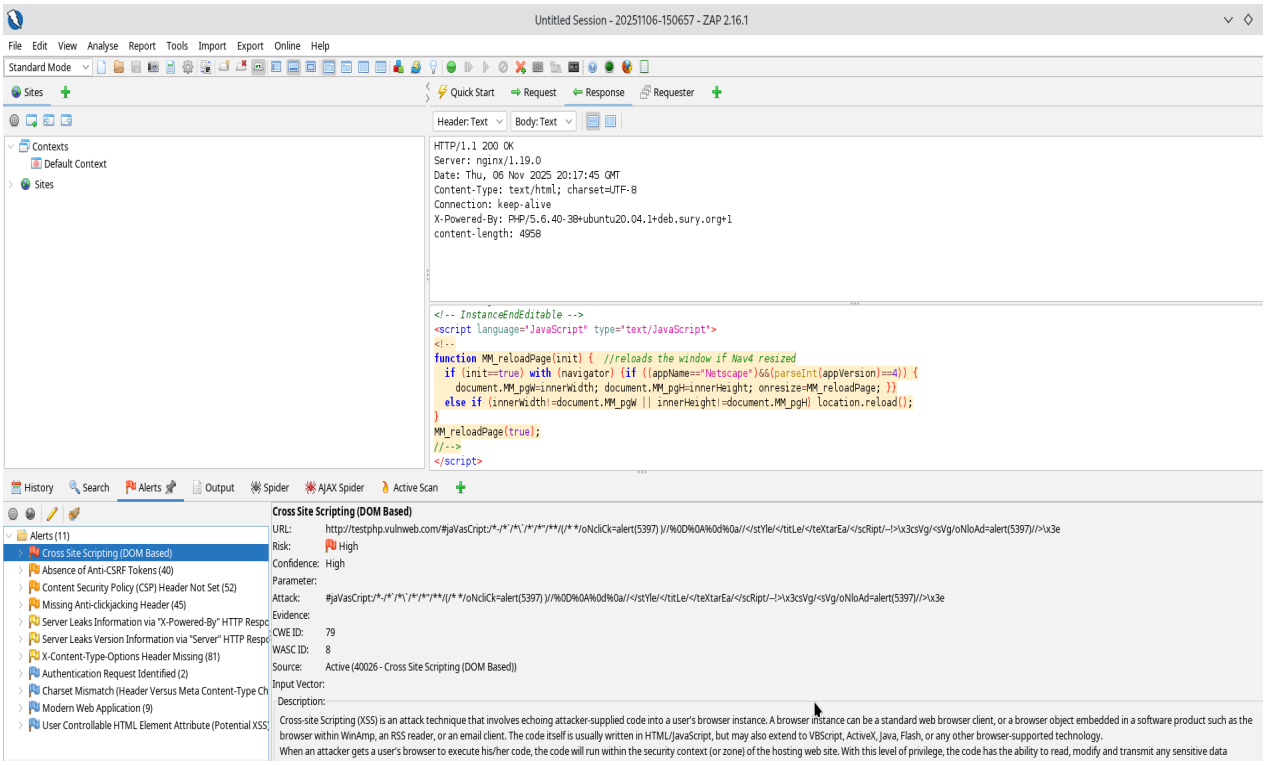

Figure 4.1: OWASP ZAP automated attack

5 Exploitation

In this phase we try to exploit identified vulnerabilities using techniques such as SQL injection, cross-site scripting, or password cracking to gain unauthorized access to systems or data. The goal is to simulate real-world attack scenarios and assess the potential damage an attacker could cause.
So in following the list of exploited vulnerabilities in this task for in our target.

5.1 SQL Injection

SQL Injection one the most common vulnerabilities in the web , by it high damage cause using inputs miss configuration and the weakness of protecting SQL queries and control , in this demo we have tried it successfully on login form .
We decide to log in without using the password by using `test' or '1'='1` as input in the username field , which makes to return the user and replace password verification by the condition `1=1` that's always correct and click login to get the user test infromation.

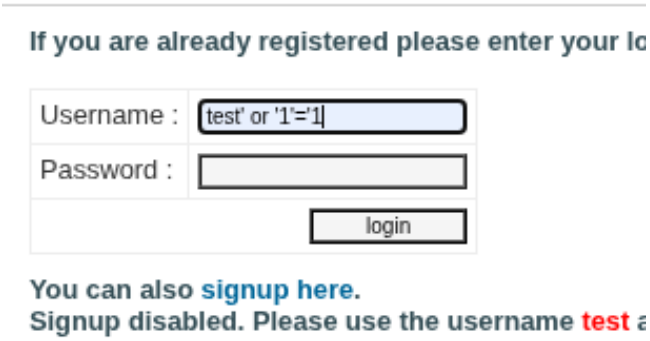

Figure 5.1: Login form input

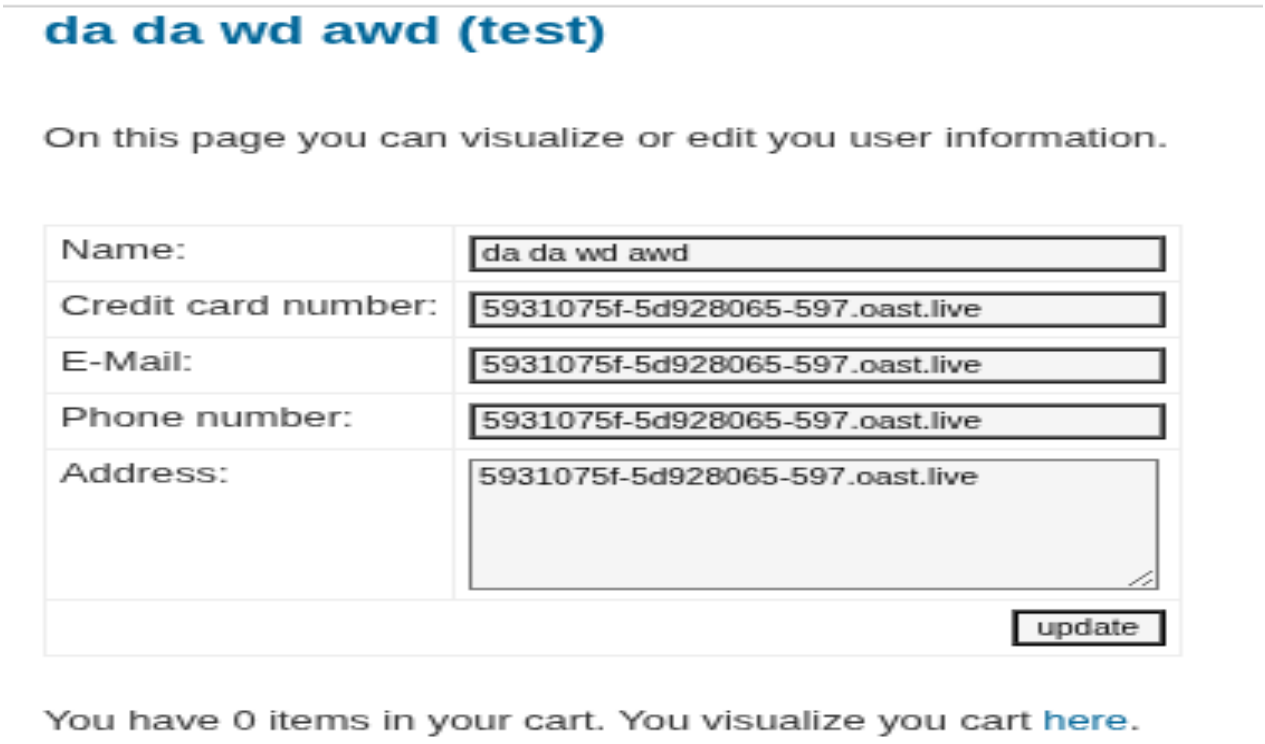

Figure 5.2: SQL Injection results

5.2 Cross-Site scripting (XSS)

For cross-site scripting is another attack related to inputs protection safty which makes an attacker execute malicious javascript code , and we found two input fields that're vulnerable to this attack , the first one is the search input in the home page , and the second exist in link `http://testphp.vulnweb.com/guestbook.php/` message input section. In the two infected field we use a test payload example : `<script> alert("XSS Found")</script>`
And the following images demonstrate that we successfully exploit this attack :

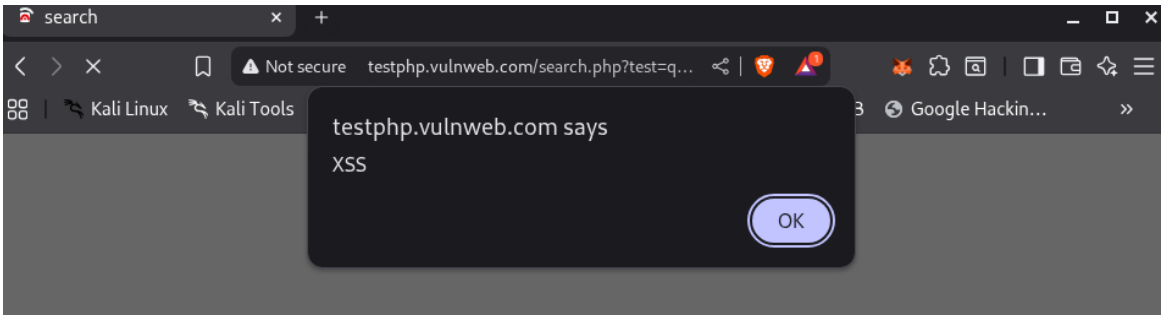

Figure 5.3: XSS in search bar

And here the demo of `guestbook.php` message input exploiting :

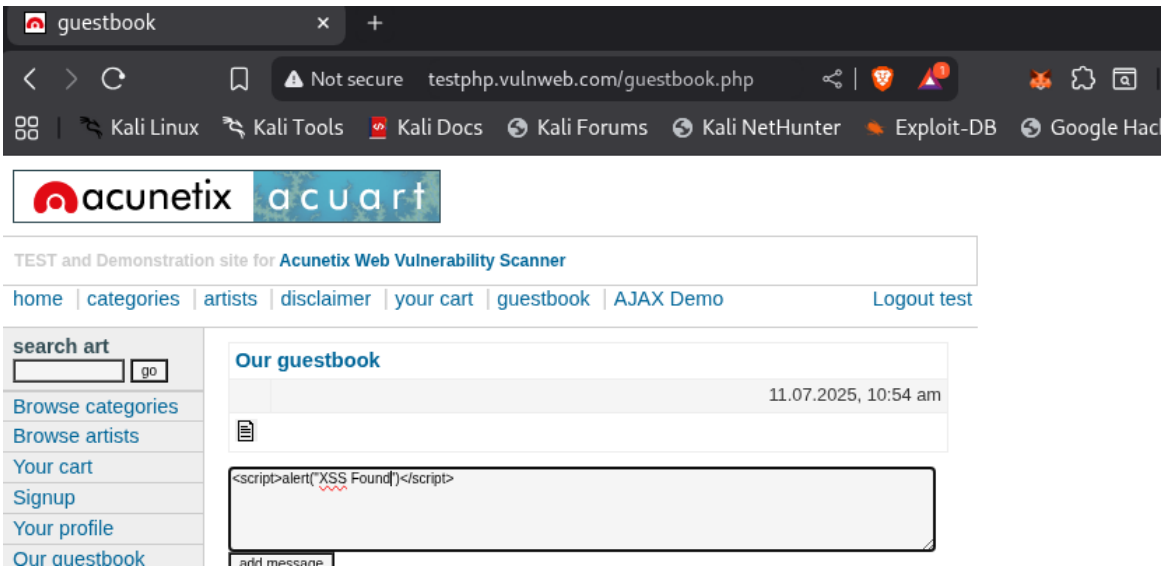

Figure 5.4: Inject JS payload as message

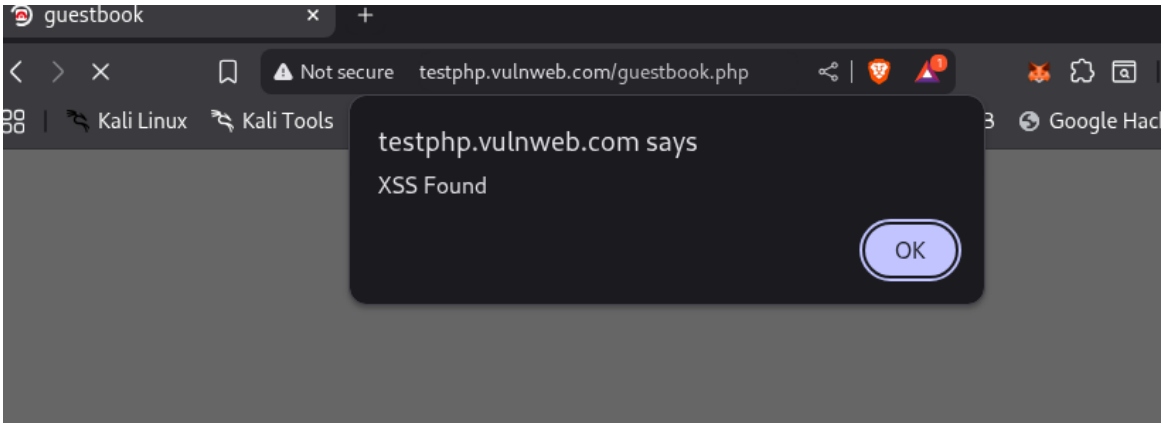

Figure 5.5: View result as alert

5.3 Local File Inclusion (LFI)

In this section we demonstrate that we can read files from local system of the target by exploiting the weakness of links that're reading files , after some research we get the link `http://testphp.vulnweb.com/listproducts.php?cat=1` which list products on the site we observe that when we click on the product image redirect us to another link : `http://testphp.vulnweb.com/showimage.php?file=./pictures/1.jpg` to display the picture by using `file` parameter.

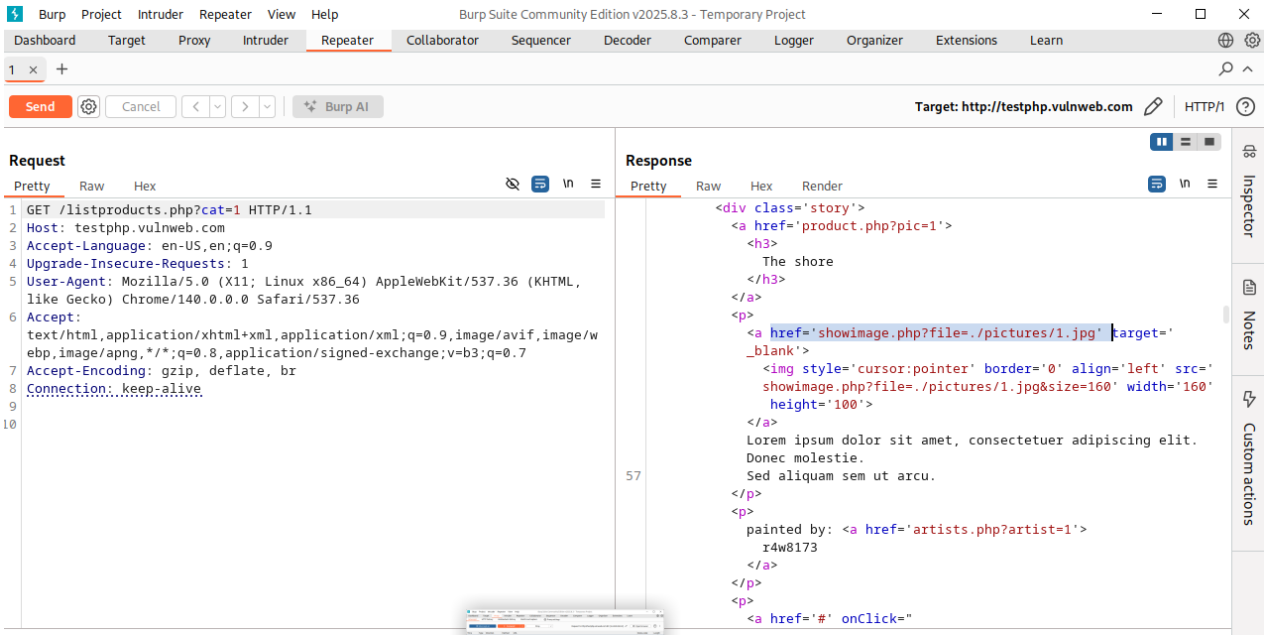

Figure 5.6: Get sensible link

An attacker can change the `file` value to get unauthorized data, as we demonstrate using BurpSuite to view response content in the next screenshots:

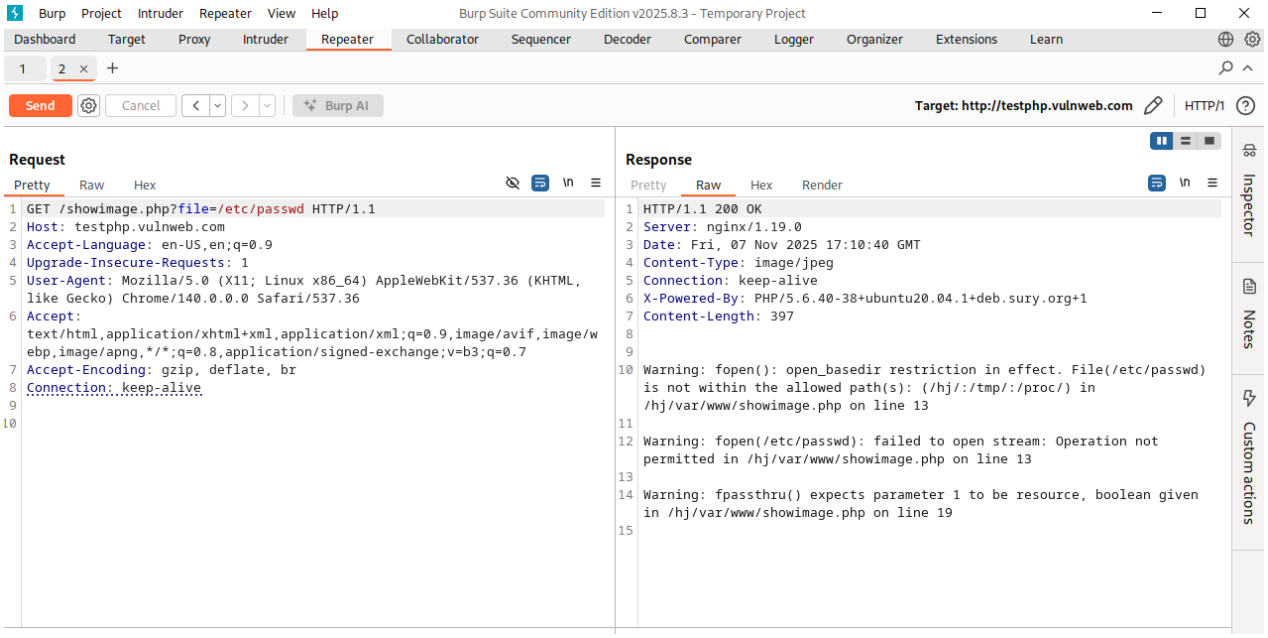

Figure 5.7: Test found link

In this picture we test to get the `/etc/passwd` sensible file in the Linux system and get user list . Request sent with success and the response is a warning which we can make just some modification on the file path , as showed in the next picture :

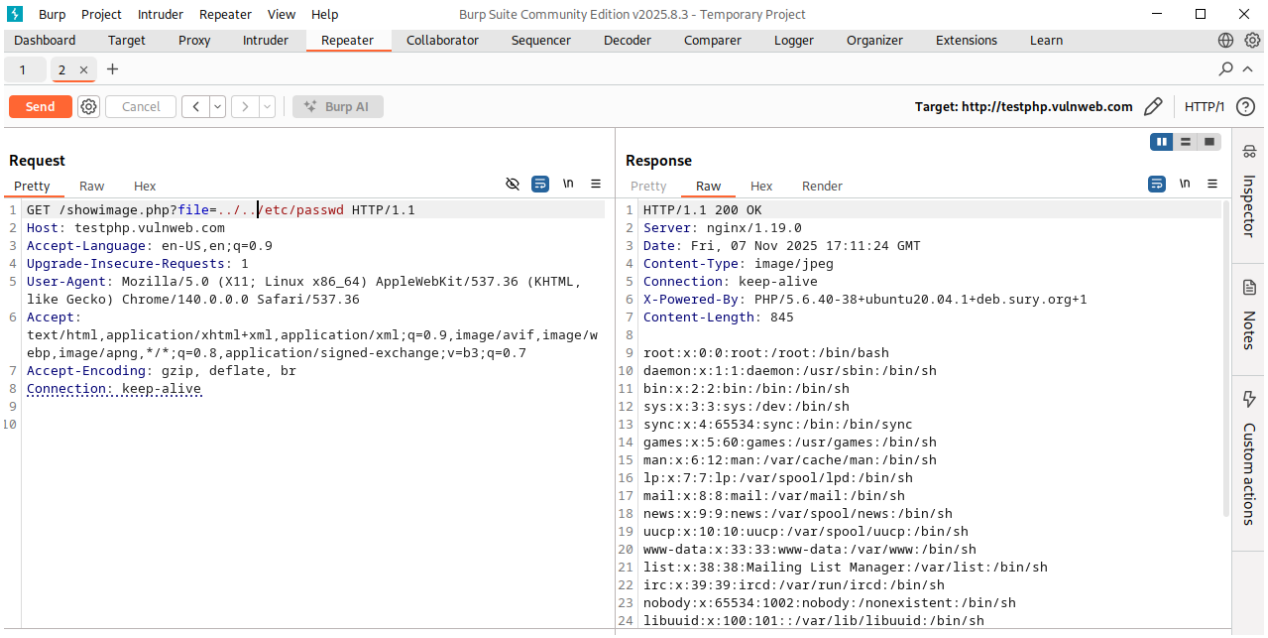

Figure 5.8: Success response of passwd file content

So we can get the `/etc/passwd` file content as the response , but this can go far to execute other malicious script or exploit weakness to damage the operating system.

6 Conclusion

In the conclusion the `testphp.vulnweb.com` has more vulnerabilities and we demonstrate the expected ones and gives our report on them. There is no safe system but by penetration testing we can find , evaluate , and correct the weakness points in the system, using the security norms, like :

- Control and protect SQL queries to avoid SQL injection attacks.
- Evaluate input fields to ignore execution of malicious JS payloads.
- Avoid using local files browsing to protect the system from the LFI attack.
- Use security testing tools by developers .
- Keep updating software technologies to the power versions.

By engaging in this scope we discover the highest vulnerabilities in the web (XSS, SQLi, LFI, RCE, ...) and benefit from there techniques to exploit them, in addition to learn more about most known kali linux tools for penetration testing.