

# Vulnerability Assessment And Penetration Testing

PREPARED FOR:

Career path orientation

Confidential

DOCUMENT VERSION CONTROL

**DOCUMENT VERSION CONTROL*****Data Classification – Client Confidential***

|                        |      |
|------------------------|------|
| <b>Client Name</b>     |      |
| <b>Project name</b>    | VAPT |
| <b>Authors</b>         |      |
| <b>Approved by</b>     |      |
| <b>Version</b>         |      |
| <b>Submission Date</b> |      |

## Section 1: Executive Summary

---

This report documents the findings after testing (irsheidat.softether.net). A series of tests were conducted against the targeted scope, using testing tools and where appropriate, manual testing techniques, to establish the presence of actual or potentially exploitable security vulnerabilities, which if exploited could result in direct or indirect damage to Customer Name. These key findings (classified as “**High**”, “**Medium**” or “**Low**”) have been highlighted in this report. Please refer to the Detailed Penetration Testing Results of the report for in depth details the key findings, their associated risks, and recommended actions.

The following represents the definition and the description of each severity rate.

| Impact        | Description  |
|---------------|--|
| <b>High</b>   | A vulnerability that can be exploited by the attacker and cause huge damage to application components and data.  |
| <b>Medium</b> | A vulnerability that can be exploited by the attacker and cause moderate damage to application components and data.  |
| <b>Low</b>    | A vulnerability that can be exploited by the attacker to understand application underlying technologies and versions which can be utilized in further attacks. |

### 1.1 Scope Details

The scope of evaluation and testing covered the following assets:

| # | Host                    | Platform |
|---|-------------------------|----------|
| 1 | Irsheidat.softether.net |          |

## Section 2: Detailed VAPT Results

---

### 2.1 Introduction

Target domain (irseidat. Softether.net)

The following is the detailed report for our findings during the Penetration testing process.

### 2.2 Restrictions

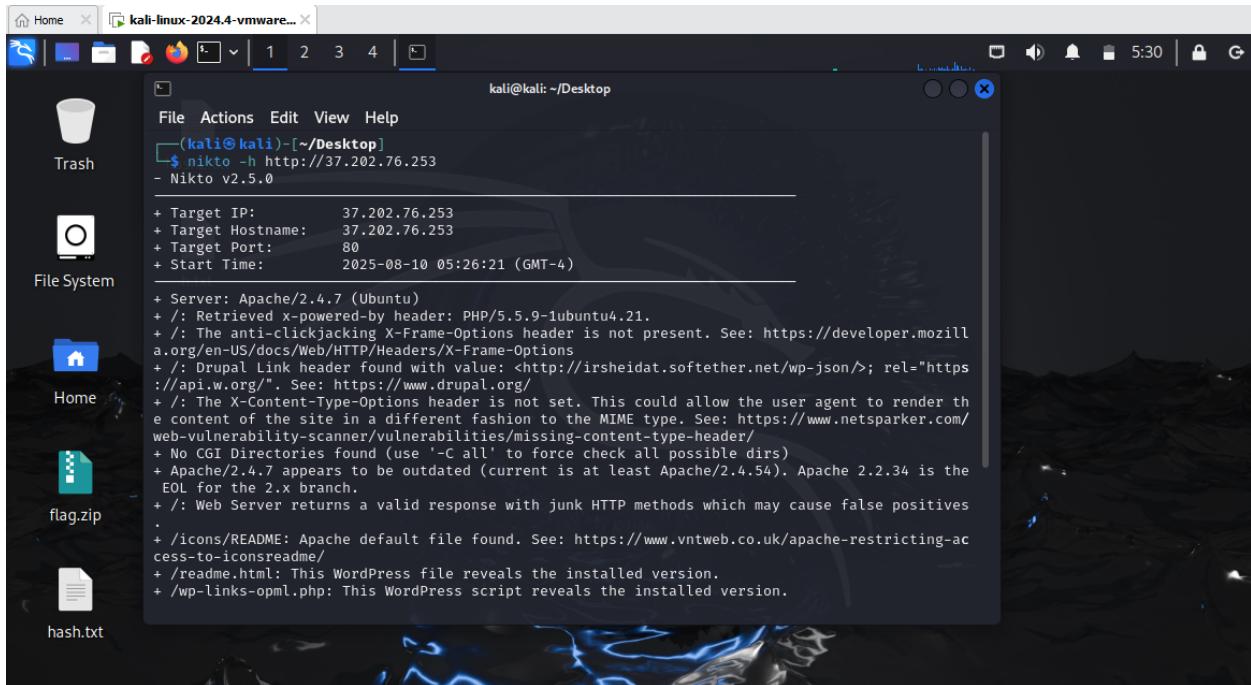
#### 1. Cookie without HttpOnly flag on the registration page:

It was found that the `wordpress_test_cookie` is created without the `HttpOnly` attribute on the registration page (`/wp-login.php?action=register`). Lack of the `HttpOnly` flag makes the cookie accessible to client-side scripts, increasing the risk of theft via Cross-Site Scripting (XSS) attacks. It is recommended to set the `HttpOnly` attribute on all sensitive cookies to prevent access by JavaScript

#### 2. Exposed WordPress login page:

The WordPress login page is accessible at `/wp-login.php` without additional protections like rate limiting or two-factor authentication (2FA). This exposure allows attackers to perform brute force attacks on user accounts. It is recommended to implement protections such as 2FA and rate limiting to mitigate this risk.

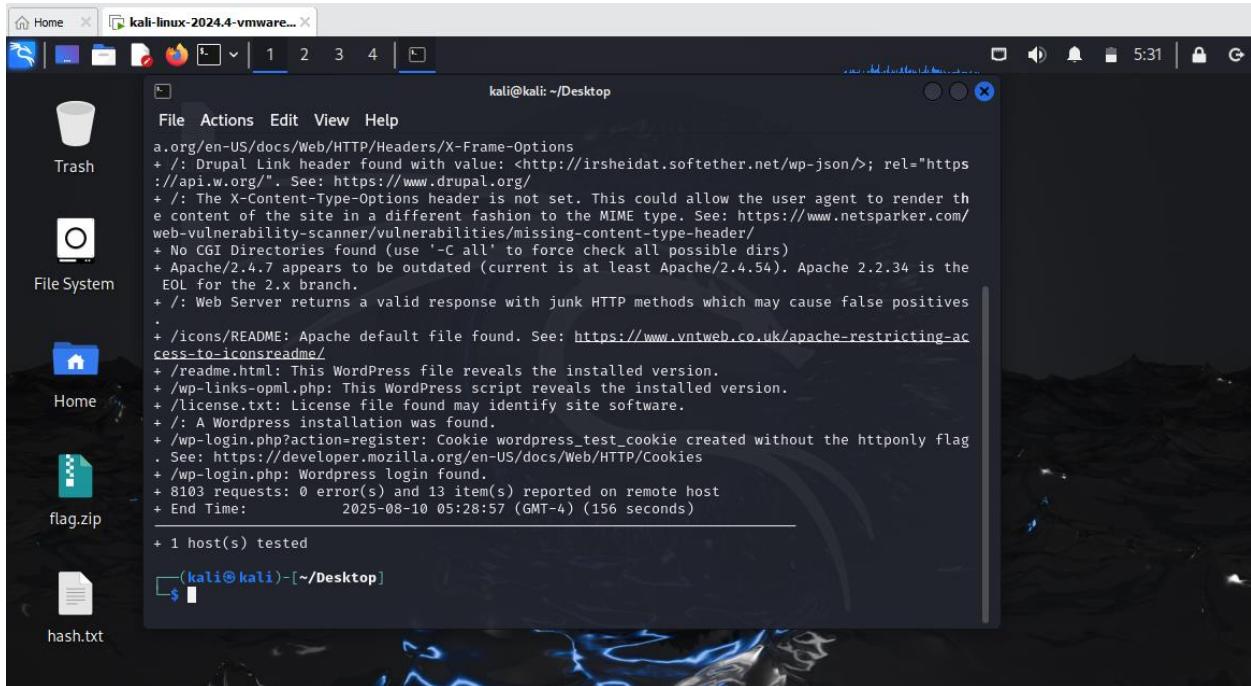




```
kali@kali: ~/Desktop
$ nikto -h http://37.202.76.253
- Nikto v2.5.0

+ Target IP:      37.202.76.253
+ Target Hostname: 37.202.76.253
+ Target Port:    80
+ Start Time:    2025-08-10 05:26:21 (GMT-4)

+ Server: Apache/2.4.7 (Ubuntu)
+ /: Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.21.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Drupal Link header found with value: <http://irsheidat.softether.net/wp-json/>; rel="https://api.w.org". See: https://www.drupal.org/
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.4.7 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives .
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconreadme/
+ /readme.html: This WordPress file reveals the installed version.
+ /wp-links-opml.php: This WordPress script reveals the installed version.
```



```
kali@kali: ~/Desktop
$ nikto -h http://37.202.76.253
- Nikto v2.5.0

a.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Drupal Link header found with value: <http://irsheidat.softether.net/wp-json/>; rel="https://api.w.org". See: https://www.drupal.org/
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.4.7 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives .
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconreadme/
+ /readme.html: This WordPress file reveals the installed version.
+ /wp-links-opml.php: This WordPress script reveals the installed version.
+ /license.txt: License file found may identify site software.
+ /: A Wordpress installation was found.
+ /wp-login.php?action=register: Cookie wordpress_test_cookie created without the httponly flag . See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
+ /wp-login.php: Wordpress login found.
+ 8103 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time:        2025-08-10 05:28:57 (GMT-4) (156 seconds)

+ 1 host(s) tested
```

## 2.3 Recommendations

1. Set HttpOnly flag for all sensitive cookies: Ensure that cookies like `wordpress_test_cookie` and any session cookies have the `HttpOnly` attribute set. This will prevent client-side scripts from accessing these cookies, reducing the risk of theft through XSS attacks.
2. Implement strong login protections: Protect the WordPress login page by enabling mechanisms such as:
  - Rate limiting to block or slow down repeated login attempts.
  - Two-factor authentication (2FA) to add an extra layer of security beyond passwords.
  - CAPTCHA or similar anti-bot measures to prevent automated brute force attacks.
  - Monitoring and alerting on suspicious login activities.
3. Regularly test for XSS vulnerabilities: Since cookies without `HttpOnly` can be stolen if XSS exists, conduct frequent security testing and code reviews to identify and fix XSS issues.
4. Consider hiding or restricting access to `/wp-login.php`: If possible, restrict access to the login page by IP whitelisting, or move it behind additional authentication layers or custom URLs to reduce exposure.

## CONCLUSION

The assessment identified medium-risk vulnerabilities related to cookie security and login page exposure on the WordPress site. The absence of the HttpOnly flag on cookies increases the risk of session theft through potential XSS attacks, while the publicly accessible login page without adequate protections leaves the site vulnerable to brute force attempts. By implementing the recommended security controls—such as setting proper cookie flags, enabling multi-factor authentication, rate limiting, and regular vulnerability testing—the overall security posture of the WordPress installation can be significantly improved, reducing the likelihood of unauthorized access and potential compromise.

**END OF THE REPORT**