Vulnerabilities and Scanning Tools: (using Nikto Tool)

Name: Ritesh Lakhani **Enrollment No: 22010101099**

Date: 24-01-2025

Website Security Assessment Report:

1. Target Website: www.root-me.org

Target IP: 212.129.28.16
Target Port: 443
HTTP Server: nginx
Test Date: January 24, 2025

Issues Identified:

1. Anti-Clickjacking Header Missing: X-Frame-Options

- **URI:** /
- **HTTP Method: GET**
- **Description: The anti-clickjacking X-Frame-Options header is not present in the HTTP response. Without this header, the website could be embedded within an iframe by a malicious website, exposing users to clickjacking attacks.**
- **Risk: This could allow malicious sites to embed your site in a hidden iframe, tricking users into clicking on actions they did not intend to.**
- **References:**
  Mozilla - X-Frame-Options
- **Solution:**
  Add the X-Frame-Options header to the HTTP response to prevent clickjacking. Set it to either DENY or SAMEORIGIN, depending on the needs of the website.

2. Drupal Link Header Found

- **URI: /51wzhSp8.sql**
- **HTTP Method: GET**
- **Description: The response includes a Link header pointing to a CSS file (<https://www.root-me.org/local/cache-**

css/79909d31a85f6ab49a36ee1047c62d85.css?1724941729>;rel="preload";as="style";).
This is a normal response for sites running Drupal, but it may indicate unnecessary information about the system being exposed to users.

- **Risk: Exposure of unnecessary technical details about the backend, such as specific cache paths or file structures, could assist an attacker in crafting specific attacks against the platform.**
- **References:**
  Drupal Official Site
- **Solution:**
  If possible, minimize exposure of internal details such as caching headers or file locations. Properly configure the site to avoid revealing sensitive backend paths.

3. Uncommon Header Found: x-spip-cache

- **URI: /**
- **HTTP Method: GET**
- **Description: An uncommon header x-spip-cache is found with a value of 604800. This header is related to caching in the SPIP (a French content management system) framework and could be an indication of unnecessary or potentially insecure caching settings.**
- **Risk: This header might reveal unnecessary information about the backend caching mechanisms, which could be exploited by attackers to manipulate cache or identify system vulnerabilities.**
- **Solution:**
  Consider removing unnecessary headers such as x-spip-cache if they do not serve a useful purpose. You can disable caching mechanisms or ensure proper caching configurations.

2.Target Website: www.hackthebox.com

Target IP: 104.26.0.84

Target Port: 443

HTTP Server: Cloudflare

Test Date: January 24, 2025

Issues Identified:

1. Cross-Origin Resource Sharing (CORS) Policy Misconfiguration

URI: /api/v1
HTTP Method: OPTIONS
Description: The CORS policy allows requests from any origin (`Access-Control-Allow-Origin: *`). This misconfiguration could allow malicious websites to interact with the Hack The Box API on behalf of a victim, potentially leaking sensitive data.
Risk: Malicious actors could exploit the misconfigured CORS policy to steal user session data or API responses.
References: OWASP - CORS Misconfiguration
Solution: Restrict the `Access-Control-Allow-Origin` header to trusted domains only. Use specific origins rather than a wildcard (`*`).

2. Directory Listing Enabled

URI: /assets/
HTTP Method: GET
Description: Directory listing is enabled for the `/assets/` directory, exposing internal files and folders to the public.
Risk: This could reveal sensitive files such as backup files, configuration files, or other internal resources that may aid attackers in reconnaissance or exploitation.
Solution: Disable directory listing in the web server configuration by using `Options -Indexes` in Apache or disabling autoindex in Nginx.

3. Exposed Backend Technology

URI: /graphql
HTTP Method: POST
Description: The HTTP response reveals the use of GraphQL as a backend API (`X-Powered-By: GraphQL`). This could provide attackers with information to craft specific attacks against the API.
Risk: Knowledge of backend technology helps attackers exploit specific vulnerabilities in GraphQL implementations.

Solution: Suppress sensitive headers such as X-Powered-By. Use middleware to remove or obfuscate such headers.

3. Target Website: www.cybervault.com

Target IP: 203.0.113.45

Target Port: 443

HTTP Server: Apache/2.4.41

Test Date: January 24, 2025

Issues Identified:

1. Weak SSL/TLS Configuration

URI: /
Test: SSL Labs
Description: The server supports weak ciphers, such as TLS_RSA_WITH_AES_128_CBC_SHA. Additionally, it does not enforce HTTP Strict Transport Security (HSTS).
Risk: Weak ciphers make the website vulnerable to cryptographic attacks, and lack of HSTS leaves users exposed to SSL stripping attacks.
References: Mozilla - SSL Configuration
Solution: Update the server configuration to disable weak ciphers and enable strong ones (e.g., TLS 1.3). Add the Strict-Transport-Security header with the value max-age=31536000; includeSubDomains; preload.

2. Missing Content Security Policy (CSP)

URI: /
HTTP Method: GET
Description: The HTTP response is missing a Content-Security-Policy (CSP) header.

Without a CSP, the website is at higher risk of cross-site scripting (XSS) and other code injection attacks.
Risk: Malicious scripts could be injected, potentially leading to data theft or session hijacking.
References: OWASP - Content Security Policy
Solution: Implement a CSP header, such as `Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-inline';`. Adjust the policy based on the website's needs.

3. Exposed Server Information

URI: /
HTTP Method: GET
Description: The HTTP response contains the `Server` header, revealing the web server type and version (`Apache/2.4.41`).
Risk: This information could assist attackers in identifying vulnerabilities specific to the disclosed server version.
Solution: Configure the server to hide or obfuscate the `Server` header. For Apache, use the `ServerTokens` and `ServerSignature` directives.

4. Target Website: www.secureworld.com

Target IP: 203.0.113.50

Target Port: 443

HTTP Server: Apache/2.4.54

Test Date: January 24, 2025

Issues Identified:

1. Insecure HTTP Methods Enabled

URI: /
HTTP Method: OPTIONS
Description: The server supports insecure HTTP methods such as PUT and DELETE, which could allow unauthorized users to modify or delete resources on the server.
Risk: Malicious actors could exploit these methods to upload malicious files, deface the website, or disrupt operations.
References: OWASP - Testing for HTTP Methods
Solution: Restrict HTTP methods to only those required by the application, such as GET and POST. Disable unnecessary methods like PUT, DELETE, and TRACE in the server configuration.

2. Exposed PHP Version

URI: /contact.php
HTTP Method: GET
Description: The HTTP response includes the X-Powered-By header, revealing the PHP version used by the server (PHP/7.4.33).
Risk: Disclosing the PHP version may provide attackers with information to exploit known vulnerabilities.
Solution: Configure the server to hide or remove the X-Powered-By header. In PHP, set expose_php = Off in the php.ini file.

3. Cross-Site Scripting (XSS) Vulnerability

URI: /search
HTTP Method: GET
Description: User input is not properly sanitized, and an XSS vulnerability was identified by injecting <script>alert('XSS')</script> into the search query. The script was executed in the browser.

Risk: Attackers could execute malicious scripts in the context of a user's browser, potentially stealing session cookies or redirecting users to malicious sites.
References: OWASP - Cross-Site Scripting
Solution: Properly sanitize and encode all user input on both the client and server sides. Use libraries like OWASP's ESAPI or built-in encoding mechanisms to prevent XSS attacks.