Advanced Vulnerability Scanner Report

Project Overview:

This Advanced Vulnerability Scanner is a command-line tool designed to identify common web application security vulnerabilities, including:

- Security Header Checks: Identifies missing HTTP headers that contribute to web application security.
- XSS Vulnerabilities: Detects possible Cross-Site Scripting (XSS) vulnerabilities by scanning forms.
- SQL Injection (SQLi) Checks: Identifies potential SQL injection points through common payloads.
- Port Scanning: Checks for open ports on the target server to determine exposed services.

Features:

- Security Header Check: Scans HTTP headers to ensure the presence of security headers like Strict-Transport-Security, Content-Security-Policy, etc.
- XSS Check: Automatically crawls internal links and checks for potential XSS vulnerabilities in forms by injecting <script>alert(1)</script> payloads.
- SQL Injection Check: Performs a simple SQL injection test using ' OR '1'='1 payloads on query parameters.
- Port Scan: Scans common ports to check for open services like HTTP, HTTPS, FTP, etc.
- Report Generation: Saves scan results in both text and JSON formats for easy parsing and further analysis.
- Multithreaded Port Scanner: Speed up port scanning by using multiple threads to scan open ports in parallel.

Dependencies:

The following libraries are required to run the scanner:

- requests: For making HTTP requests.

- beautifulsoup4: For parsing HTML content and crawling links.

- colorama: For adding color to the output in the terminal.

- socket: For checking open ports.

To install dependencies:

pip install -r requirements.txt

Installation:

1. Clone the repository or download the files.

2. Install dependencies:

   pip install -r requirements.txt

3. Run the scanner with the following command:

   python vuln_scanner.py --url http://example.com --headers --xss --sqli --ports --json report.json

   Replace http://example.com with the target website you want to scan.

Command-Line Arguments:

--url <URL>: Specify the target URL (e.g., http://example.com).

--headers: Run the security header check.

--xss: Run the XSS vulnerability check.

--sqli: Run the SQL injection check.

--ports: Scan for open ports on the target host.

--report <filename>: Specify the output filename for the text-based report (e.g., scan_report.txt).

--json <filename>: Specify the output filename for the JSON-based report (e.g., scan_report.json).

Sample Output:

Security Header Check:

[+] Checking security headers...

[[PASS]] Strict-Transport-Security is present

[[PASS]] X-XSS-Protection is present

[!] Content-Security-Policy is missing

[[PASS]] X-Frame-Options is present

[[PASS]] X-Content-Type-Options is present

XSS Check:

[+] Checking for XSS on internal forms...

[[PASS]] No XSS vulnerability in form at http://example.com/contact

[!] Possible XSS in form at http://example.com/login

SQL Injection Check:

[+] Checking for SQL Injection...

[[PASS]] No SQL injection found at http://example.com?id=' OR '1'='1

[!] Possible SQLi vulnerability at http://example.com?id=' OR '1'='1

Port Scan:

[+] Scanning common ports (multi-threaded)...

[[PASS]] Port 80 (HTTP) is OPEN

[[PASS]] Port 443 (HTTPS) is OPEN

[[PASS]] Port 21 (FTP) is CLOSED

JSON Report Example:

{

```
  "target": "http://example.com",
  "headers": [
    "[[PASS]] Strict-Transport-Security is present",
    "[[PASS]] X-XSS-Protection is present",
    "[!] Content-Security-Policy is missing",
    "[[PASS]] X-Frame-Options is present",
    "[[PASS]] X-Content-Type-Options is present"
  ],
  "xss": [
    "[[PASS]] No XSS vulnerability in form at http://example.com/contact",
    "[!] Possible XSS in form at http://example.com/login"
  ],
  "sqli": [
    "[[PASS]] No SQL injection found at http://example.com?id=' OR '1'='1",
    "[!] Possible SQLi vulnerability at http://example.com?id=' OR '1'='1"
  ],
  "ports": [
    "[[PASS]] Port 80 (HTTP) is OPEN",
    "[[PASS]] Port 443 (HTTPS) is OPEN",
    "[[PASS]] Port 21 (FTP) is CLOSED"
  ]
}
```


Future Improvements:

1. Advanced XSS Detection: Implement more sophisticated payloads for testing a wider range of XSS vulnerabilities.

2. Custom SQLi Payloads: Expand the SQLi check with more advanced and custom payloads.

3. Reporting Improvements: Enhance the JSON report with additional metadata such as timestamp and scan duration.

4. Authentication Support: Implement features for handling websites with authentication (basic auth, cookies, etc.).

Conclusion:

This tool provides a simple yet effective way to perform web application security checks for common vulnerabilities. It is designed to be easy to use and extendable for more advanced vulnerability detection in the future.