USER GUIDE FOR WEBGUARD FRAMEWORK

1. Introduction

This user guide explains how to install, configure, and use the Penetration Testing Framework for Web Applications. This framework automates common security testing tasks, including vulnerability scanning, security header checks, SSL/TLS analysis, and active attack simulation using integrated modules and third-party tools.

2. System Requirements

Before running the framework, ensure that the following system requirements are met:

Hardware

- **Processor**: Dual-core processor or higher
- **Memory**: Minimum 4 GB RAM (8 GB recommended)
- **Disk Space**: At least 500 MB of free space

Software

- **Operating System**: Linux-based systems (e.g., Ubuntu, Debian, Kali Linux)
- **Python Version**: Python 3.8 or higher
- **Package Manager**: pip or pip3
- **Dependencies**:
    1. **Python libraries**: requests, pdfkit, PyPDF2, zapv2
    2. **WhatWeb**: Ensure installed and accessible in the system path
    3. **ZAP (OWASP ZAP)**: Running on localhost or a configured server
    4. **sqlmap**: Installed and in the system path for SQL injection detection
- **Web Browser**: Any modern web browser for viewing reports

3. Installation

1. Clone the Repository

```
git clone https://github.com/alexander47777/WebGuard.git
cd WebGuard
```

2. Install Python Dependencies

```
pip install -r requirements.txt
```

3. Install External Tools

Ensure that WhatWeb, OWASP ZAP, and sqlmap are installed on your system.

- **WhatWeb:**

```
sudo apt-get install whatweb
```

- **OWASP ZAP: Download and install OWASP ZAP and start it:**
- **sqlmap**:

```
sudo apt-get install sqlmap
```

4. Configure pdfkit for Report Generation

- **Install wkhtmltopdf:**

```
sudo apt-get install wkhtmltopdf
```

4. Configuration

Configuration File (config/default_config.yaml)

Customize the framework's behavior using the configuration file. Below is an example configuration:

```
scanner:
  sql_injection: true
  xss: true
  csrf: false
  auth_bypass: true
  custom_headers:
    User-Agent: "WebAppPenTestFramework"
    Accept-Language: "en-US"

attack_engine:
  timeout: 5
  retries: 2
  payloads:
    sql_injection: ["' OR '1'='1", "' OR 'a'='a"]
    xss: ["<script>alert('XSS')</script>"]

reporting:
  output_format: "pdf"
  report_path: "./results/"
  include_details: true

zap:
  api_key: "YOUR_ZAP_API_KEY"
  url: "http://localhost:8080"
```

Modify the settings to match your specific needs and ensure that the api_key for OWASP ZAP matches the key in your ZAP instance.

5. Running the Framework

1. Basic Command

Run the main script with a target URL:

```
python3 main.py https://targetwebapp.com --config config/default_config.yaml
```

6. Viewing and Interpreting Reports

Generated Report Structure

- **WhatWeb Results**: Shows server and technology details.
- **SSL/TLS Check**: Indicates certificate status and protocol security.
- **Header Analysis**: Details security headers and their configurations.
- **Vulnerability Scan Results**: Summarizes detected vulnerabilities.
- **Attack Results**: Lists any successful or failed simulated attacks.

Accessing Reports

Reports are saved in the **results** folder in PDF format.

7. Troubleshooting and Common Issues

Issue 1: WhatWeb Not Found

- Ensure WhatWeb is installed and available in your PATH.

Issue 2: Connection Refused for ZAP

- Verify that OWASP ZAP is running on the specified host and port.

Issue 3: SSL/TLS Module Errors

- Ensure the target supports SSL/TLS or check for network issues.