Vscraper

Ameni Hsini BA/IT

Introduction

With the increasing number of cyberattacks and security breaches, ensuring the security of web applications has become a crucial task.
To address these security concerns, we have developed a web application vulnerability scanner Vsraper. Our scanner is designed to automatically detect security vulnerabilities in web applications and provide detailed reports on the identified issues.

In the following sections, we will provide a detailed overview of Vscraper, including its design, components, and functionality.

Purpose and intended users

The purpose of Vscraper is to identify security vulnerabilities or weaknesses in a web application. It does so by scanning the web application for known vulnerabilities such as SQL injection, cross-site scripting, or insecure server configuration. By identifying these vulnerabilities, the scanner helps website owners and administrators to better secure their web applications against potential attacks and data breaches.

It can also be used by security professionals to evaluate the security of third-party websites and applications. This can be particularly useful in cases where companies are considering integrating third-party software or web services into their own systems. By scanning these services, businesses can ensure that they meet the necessary security requirements and do not introduce new vulnerabilities into their own systems.

This tool is particularly useful for web developers, security researchers, and system administrators who are responsible for maintaining the security of web applications.

Features

Automated scanning

The scanner automatically identifies and assesses vulnerabilities within web applications or websites.

Vulnerability reporting

The scanner generates reports that include detailed information about vulnerabilities

Customizable scan.

Users can choose which types of vulnerabilities they want the scanner to check for.

Integration with security tools

The scanner can be integrated with other security tools, such as firewalls and intrusion detection systems, to provide a more comprehensive security solution.

Functional flows

1. **user inputs the target website URL**

2. **get_website_info(url):**

   This function sends a GET request to the website and extracts relevant information about the server type and URLs used.

3. **search_vulnerabilities(info):**

   This function takes the website information retrieved by **get_website_info** as input and searches for vulnerabilities in a vulnerability database, then extracts any identified vulnerabilities.

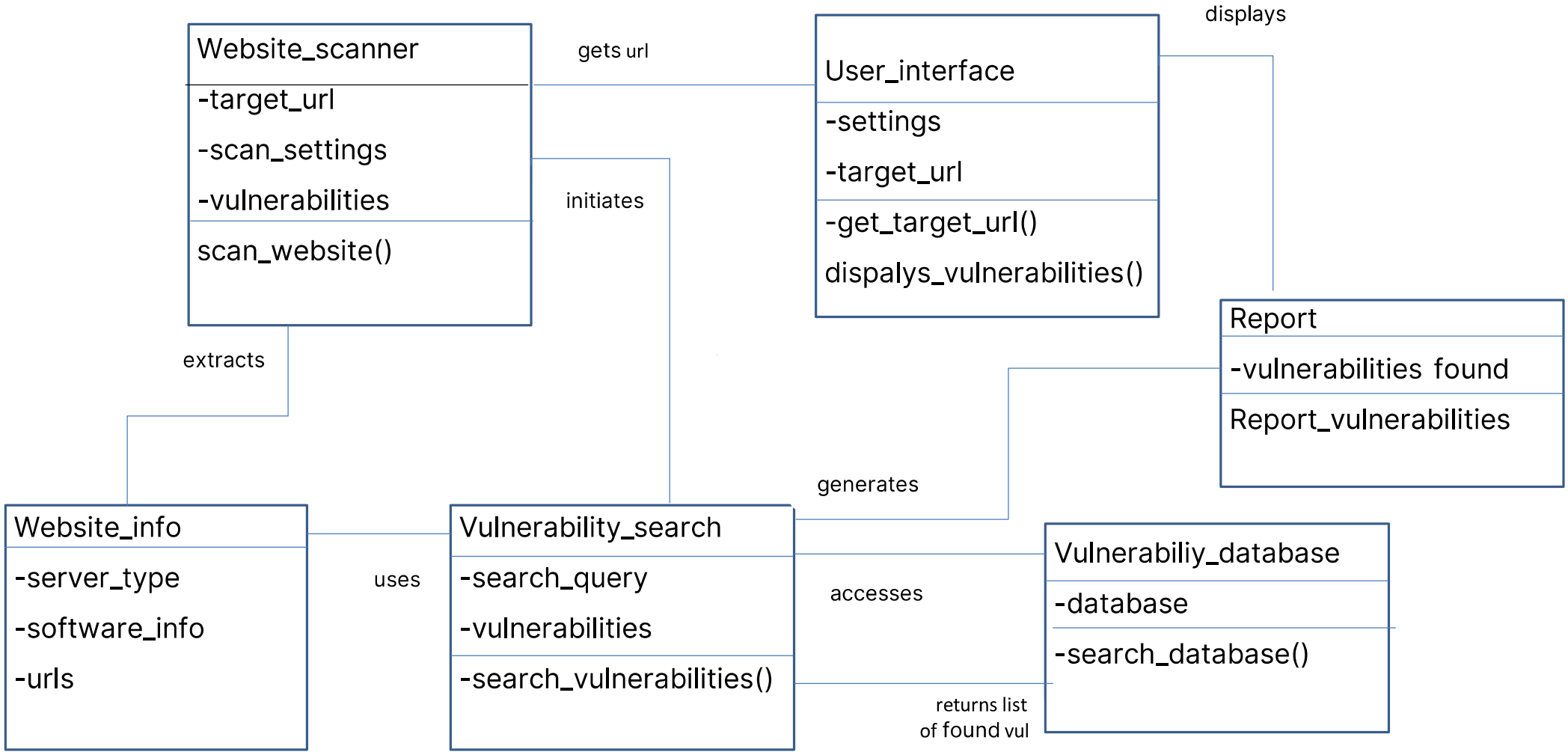
3. **report_vulnerabilities(url, vulnerabilities):**

This function takes the website URL and identified vulnerabilities as input and reports the vulnerabilities to the user. It does this by printing the list of identified vulnerabilities for the website if found.

4. **scan_website(url):**

This function is the main function that ties everything together. It takes a website URL as input and calls **get_website_info** to retrieve information about the website. It then calls **search_vulnerabilities** to search for vulnerabilities in the retrieved information. Finally, it calls **report_vulnerabilities** to report the identified vulnerabilities to the user..

Class diagram



Development phases

1. Planning

This phase involved the identification of the requirements and scope for vulnerability web scraper as well as research on existing vulnerability scanners and tools. We also collected information on common vulnerabilities and the methods used to identify them.

2. Design

In this phase, we created a design document for the scanner, which included the overall architecture, system's input and output, user interface design, and detailed descriptions of the individual functions and modules.

- **Tools used** :
  - Draw.io for system architecture design

3. Implementation

In this phase, Vscraper was developed via these three main functions: **get_website_info(url), search_vulnerabilities(info), report_vulnerabilities(url, vulnerabilities),**

- **Tools used:**
  - Python programming language.
  - Requests: Requests is a Python library that is used to send HTTP requests and handle responses. It was used in this project to send requests to the website being scanned and to the vulnerability database.
  - The National Vulnerability Database (NVD) provided by the National Institute of Standards and Technology (NIST). The URL for the vulnerability search query is: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query= it is used to search for known vulnerabilities in the software and server versions used by the website being scanned.

4. User interface design

In this phase, we created the user interface for the scanner. The user interface included a form for the user to input the website URL, a button to initiate the scan, and a results page that displayed the identified vulnerabilities.

- **Tools used:**
  - Flask: Flask is a lightweight web application framework that was used to develop the user interface for the scanner.
  - HTML/CSS for web interface design

5. Testing

In this phase, the scanner was tested to ensure that it identified vulnerabilities accurately and reliably.

- **Tools used** : web applications containing multiple vulnerabilities designed as a training tool for web security testing.