Security Vulnerability Scanner & information gathering tool

1). 1ntroduction:

This project is a vulnerability scanner and information gathering tool designed in cybersecurity. It runs on Kali Linux, allowing users to scan their systems for vulnerabilities and gather intelligence on potential targets. With a user-friendly interface and step-by-step guidance, it requires no prior cybersecurity knowledge, making it an excellent tool for learning and securing systems.

Purpose:

The primary purpose of this tool is to provide an automated way to identify security weaknesses in a network or system. It helps users perform penetration testing, discover open ports, analyze running services, and detect vulnerabilities that could be exploited by attackers.

Objectives:

1.Enable beginners to learn and practice cybersecurity skills.

2.Provide an easy-to-use interface for security scanning.

3.Automate vulnerability scanning to improve system security.

4.Help organizations and individuals identify security risks before they can be exploited.

Importance in Real-World Applications:

1.Detect vulnerabilities before malicious attackers exploit them.

2.Improve the security posture of organizations by identifying misconfigurations.

3.Conduct penetration testing in ethical hacking environments.

4.Learn about cybersecurity methodologies through practical application.

•……•

2). Project Scope:

Features and Functionalities:

The Vulnerability Scanner and Information Gathering Tool is designed to provide a comprehensive and beginner-friendly approach to scanning systems for vulnerabilities and gathering useful information. The key features include:

- **TCP SYN Ping** for detecting live hosts behind stateful firewalls.
- **DNS Enumeration via Dnsenum** to extract DNS information and discover non-contiguous IP blocks.
- **Nmap Integration** for scanning open ports and determining the operating system and version of a host.

- **Service/Version Probing** using Nmap to identify specific software vulnerabilities tied to service versions.
- **Vulners Script** to identify known vulnerabilities (CVEs) by comparing discovered services against the **vulners.com** vulnerability database.
- **OWASP ZAP** integration to automatically find and report security vulnerabilities in web applications.
- **OWASP Nettacker** automation for information gathering, vulnerability scanning, and report generation.
- **Scan Results Storage** in the **my_reports folder** for easy access and analysis of scan results.
- **Program Clean-up Function** to ensure all components, folders, and reports are deleted after use.

Target Users or Systems:

The tool is primarily designed for beginners in the field of cybersecurity, making it accessible for those who are:

- **New to Cybersecurity**: Users who have little to no prior knowledge of security concepts or terminal-based operations.
- **System Administrators**: Those looking to quickly assess the security posture of their systems.
- **Penetration Testers**: Both beginners and intermediate pentesters seeking an automated way to gather information and identify vulnerabilities in a target system or network.
- **Developers**: Particularly web application developers who can use the integrated OWASP ZAP feature to test their applications for common vulnerabilities.

•……•

3). Technologies Used:

Kali Linux:

Kali Linux is a specialized Linux distribution designed for penetration testing and security tasks. It offers pre-installed tools for vulnerability scanning, network analysis, and exploitation, making it an ideal platform for security projects. Its lightweight and customizable nature also supports seamless integration with scanning tools.

Python Programming Language:

Python is highly suited for security vulnerability projects due to its simplicity, extensive libraries, and versatility. It enables quick development of scripts for scanning, automation, and data analysis. Libraries like “socket”, “subprocess”, and “nmap” streamline network and vulnerability scanning tasks. Python's widespread community support further aids in addressing project-specific challenges.

•……•

4). Operating System Concepts Demonstrated:

Security: Demonstrates security practices and vulnerability detection to ensure system integrity.

Process Management: Involves managing system processes to analyze and mitigate potential security threats.

System Vulnerability Analysis: Analyzes system weaknesses and provides insights for improvement, showcasing OS-level vulnerability assessment.

•……•

5). Implementation Details:

- **Module Imports:** It utilizes essential modules such as `os`, `sys`, `subprocess`, `shutil`, and `datetime` to handle system-level operations, interact with the shell, manage files, and work with date and time.
- **Color-Coding**: The script uses ANSI escape sequences to colorize terminal output (e.g., success messages, errors, and prompts) for better user interaction.
- **Root Privileges**: The program checks if it is being run with root privileges (using `os.geteuid() == 0`) since some commands require elevated permissions.
- **Report Storage**: It organizes the generated scan reports in a folder named `my_reports`, allowing users to save results for later review.
- **Multiple Tools Integration**: The script integrates with multiple scanning tools like `nmap`, `OWASP ZAP`, and `OWASP Nettacker` to perform security scans.

  How Modules Interact:

  The script follows a structured approach where different modules interact to perform vulnerability scanning and information gathering. It begins with a root check to verify if the user has the necessary administrative privileges, as certain security scanning tools require elevated permissions to function correctly. Command execution is primarily handled through the subprocess module, which allows the script to run external tools like nmap, OWASP ZAP, and OWASP Nettacker. For instance, when performing an nmap scan, the script executes the relevant command using subprocess.run(), captures the output, processes the results, and displays them in a structured format.

  After executing a scan, the script provides an option to save the results in the `my_reports` folder, ensuring that users can access and analyze their findings later. For web application security testing, the script integrates Docker and OWASP ZAP. It checks if Docker is installed and, if not, installs it before running OWASP ZAP in a containerized environment to perform automated security assessments on target URLs. Similarly, for network and system vulnerability scanning, the script incorporates OWASP Nettacker by checking if the tool is installed. If it is not, the script clones the Nettacker repository from GitHub, installs the necessary dependencies, and executes scans on the specified target. This structured workflow ensures that each module interacts seamlessly, providing users with an efficient and user-friendly security scanning tool.

After running our project we have a menu screen containing all features of my project and we can choose any to run.

Menu:

0: Help

1: Check availability

2: DNS enumeration

3: OS detection

4: Service and version detection

5: Vulnerability check with vulners

6: OWASP ZAP

7: OWASP Nettacker

8: View reports

9: Exit program

10:Remove program

•…….•

6). Conclusion:

In conclusion, the vulnerability scanner and information gathering tool is designed to simplify security assessments for beginners by automating complex tasks and providing a user-friendly interface. It efficiently integrates tools like `nmap`, OWASP ZAP, and OWASP Nettacker, ensuring seamless execution through `subprocess.run()`. The script handles root privilege checks, automates dependency installations, and organizes scan results in a structured manner. By leveraging Docker for web security testing and systematically managing scan execution, it enables users to identify vulnerabilities with minimal effort. This tool serves as a practical solution for security assessments, making vulnerability detection accessible to all users.

•…….•

The **Help** option provides users with an overview of the tool's functionalities and how to use each feature effectively. When selected, it displays detailed descriptions of the available scanning and security assessment options, including usage instructions and recommended configurations. This feature ensures that even users with minimal cybersecurity knowledge can navigate the tool efficiently, making it beginner-friendly.

1: Check Availability

The **Check Availability** feature helps determine whether a target system or website is accessible. It typically uses methods like **ping requests** or **basic HTTP requests** to verify if a server or IP address is

responsive. This step is essential before conducting deeper scans, as it ensures the target is online and reachable. If the target does not respond, further security assessments may not be possible.

2: DNS Enumeration

DNS Enumeration is a reconnaissance technique used to gather information about a target's domain, such as subdomains, mail servers, and name servers. The tool queries DNS records (A, MX, TXT, CNAME, etc.) to identify potential attack surfaces. Techniques like brute-forcing subdomains or querying public DNS databases help uncover hidden domains, which can be useful for penetration testing or security audits.

3: OS Detection

OS Detection attempts to identify the operating system running on a target machine. This is achieved by analyzing network response patterns, such as **TTL values, TCP window sizes, and specific packet responses**. Knowing the OS helps attackers or security professionals determine vulnerabilities specific to that system and choose the right exploits or security measures accordingly.

4: Service and Version Detection

This feature scans a target machine to **identify running services and their versions**. It helps security analysts understand the technologies being used on a system (e.g., Apache, MySQL, OpenSSH) and check whether any of them have known vulnerabilities. This is crucial for penetration testing, as outdated or misconfigured services can be exploited by attackers.

5: Vulnerability Check with Vulners

This module integrates the **Vulners database**, which contains a vast repository of known security vulnerabilities (CVEs). The tool scans a target system, matches detected services and software versions against the Vulners database, and reports if any known vulnerabilities exist. This automated vulnerability assessment helps in identifying and mitigating potential security threats before they are exploited.

6: OWASP ZAP

OWASP ZAP (Zed Attack Proxy) is an open-source web application security scanner. It is used to find vulnerabilities in web applications, such as **SQL injection, XSS (Cross-Site Scripting), and misconfigurations**. When selected, this option runs ZAP to analyze web application security, providing insights into weaknesses that could be exploited by attackers. It is widely used by security professionals and ethical hackers.

7: OWASP Nettacker

OWASP Nettacker is an automated security scanner used for network and web application security assessments. It performs **reconnaissance, scanning, and vulnerability detection** by gathering data about a target's infrastructure. This module helps identify weak points in a system, making it useful for ethical hackers, penetration testers, and organizations looking to strengthen their cybersecurity.

8: View Reports

The **View Reports** feature allows users to see previously conducted scans and their results. These reports provide details on vulnerabilities found, system configurations, and security weaknesses. Having a well-structured report helps security teams assess risks, prioritize security fixes, and maintain a log of past assessments for compliance and auditing purposes.

9: Exit Program

Selecting **Exit Program** terminates the tool's execution and closes the interface. This ensures that no background processes remain running and that system resources are freed up after scanning operations.

10: Remove Program

The **Remove Program** option is designed to uninstall the security scanner from the system. It deletes all related files, configurations, and reports to ensure no traces remain. This is useful when the tool is no longer needed or when a fresh installation is required. Some implementations may also include an option to remove dependencies installed alongside the program.

•……