Report By-
S M RAIHAN KABIR
Date: 1st March 2024

Python Script Functionality Overview

Introduction

The provided Python script offers functionality to assess the security posture of a website or IP address. It conducts checks on security headers and open ports, generating a detailed vulnerability report in PDF format.

Features

1. **Security Headers Check**
   - Retrieves security headers from the target website.
   - Identifies missing security headers.
   - Provides explanations of the importance of each security header.

2. **Open Ports Check**
   - Scans common ports of the target host to identify open ports.
   - Provides descriptions of vulnerabilities associated with each open port.

3. **PDF Report Generation**
   - Generates a comprehensive vulnerability report in PDF format.
   - Includes details on the target host, security headers, missing security headers, and open ports.
   - Presents information in a structured and readable format.

Implementation Details

- **Libraries Used:**
  - `requests`: For making HTTP requests to retrieve security headers.
  - `socket`: For scanning open ports on the target host.
  - `urllib.parse`: For parsing the input URL.
  - `reportlab`: For generating PDF reports.

- **Security Headers Check:**
  - Utilizes `requests.head()` method to retrieve HTTP headers.
  - Compares obtained headers with a predefined list of security headers.
  - Identifies missing headers and provides relevant explanations.

- **Open Ports Check:**

- Utilizes socket connections to check the status of common ports.
- Matches open ports with a predefined dictionary of associated vulnerabilities.
- Provides vulnerability descriptions for open ports.

- **PDF Report Generation:**
  - Utilizes `reportlab` library to create PDF documents.
  - Constructs the report with information on the target host, security headers, and open ports.
  - Formats the report with appropriate headings, styles, and colors.

Usage

1. **Input Prompt:**
   - Prompts the user to enter the URL or IP address of the target host.

2. **URL Validation:**
   - Validates the user input and ensures it starts with either "http://" or "https://".

3. **Execution:**
   - Initiates the security headers check and open ports scan.
   - Generates a PDF report summarizing the findings.

4. **Output:**
   - Displays the target host, identifies security headers, missing headers, and open ports during execution.
   - Prints a message confirming the generation of the PDF report.

Security Headers Check

Methodology

- Request Headers Retrieval: Utilizes the requests.head() method to retrieve HTTP headers from the target host.
- Comparison: Compares the obtained headers with a predefined list of security headers.
- Identification: Identifies missing security headers by analyzing the presence of each header in the response.
- Explanation: Provides detailed explanations for missing security headers, highlighting potential security risks associated with their absence.

Importance

- Prevention of Attacks: Security headers play a vital role in mitigating various common web attacks such as Cross-Site Scripting (XSS), Clickjacking, and Data Injection.

- Enhanced Security: Properly configured security headers help bolster the overall security posture of web applications by enforcing security policies and best practices.

Open Ports Check

Methodology

- Port Scanning: Utilizes socket connections to scan common ports on the target host.
- Port Status: Determines the status of each port, identifying whether it is open or closed.
- Vulnerability Mapping: Matches open ports with a predefined dictionary of associated vulnerabilities.
- Vulnerability Description: Provides detailed descriptions of vulnerabilities associated with each open port, elucidating potential risks and attack vectors.

Importance

- Network Security Assessment: Open port scanning is crucial for identifying potential entry points for attackers and assessing the security of network configurations.
- Vulnerability Mitigation: Understanding vulnerabilities associated with open ports enables proactive mitigation efforts, such as applying patches, implementing access controls, or employing network security solutions.

Dependencies

From terminal need to install mentioned packages with below commands:

1. **requests**: Used for making HTTP requests to retrieve security headers.
   ```
   **pip install requests**
   ```

2. **reportlab**: Utilized for generating PDF reports.
   ```
   **pip install reportlab**
   ```

These commands will install the necessary packages for the script to execute successfully. Once you've installed these packages, you should be able to run the script without any issues.

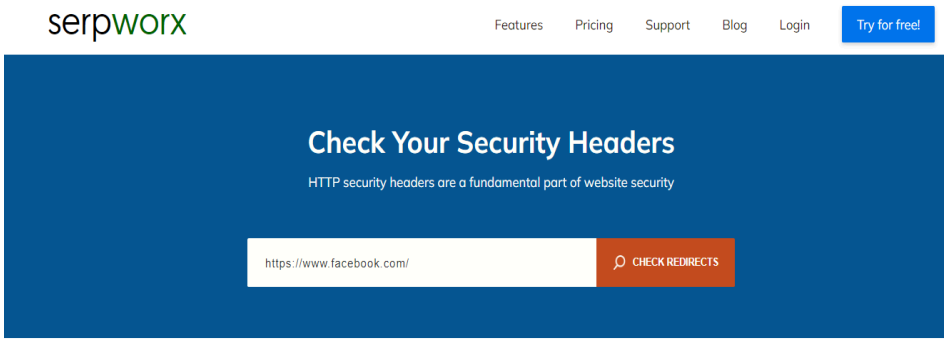
Conclusion

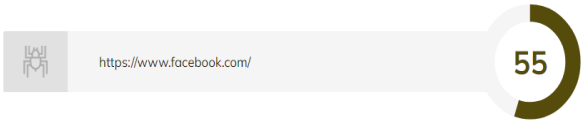
The Python script provides a convenient and systematic approach to assess the security vulnerabilities of a web application or server. By generating detailed reports, it assists security professionals in identifying and addressing potential risks effectively.

Output Comparison

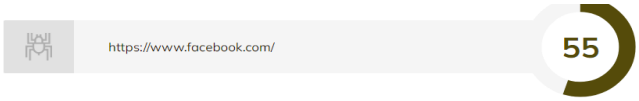
Target Host: https://www.facebook.com/

Existing Solution: https://www.serpworx.com/





Security Headers Information:



SECURITY HEADERS	HEADER RESPONSE	
X Frame Options	DENY,DENY	✓
X XSS Protection	0,0	✓
X Content Type Options	nosniff,nosniff	✓
X Permitted Cross Domain Policies	--	✗
Strict Transport Security	max-age=15552000; preload,max-age=15552000; preload	✓
Content Security Policy	default-src data: blob: 'self' https://*.fbsbx.com 'unsafe-inline' *.facebook.com ...	✓
Referrer Policy	--	✗
Feature Policy	--	✗
Expect CT	--	✗

Nmap tool Information for Open Port :

```
[ blackarch ~ ]# nmap -sV www.facebook.com
Starting Nmap 7.93 ( https://nmap.org ) at 2024-02-29 16:00 UTC
Nmap scan report for www.facebook.com (163.70.144.35)
Host is up (0.0085s latency).
Other addresses for www.facebook.com (not scanned): 2a03:2880:f10c:83:face:b00c:
0:25de
rDNS record for 163.70.144.35: edge-star-mini-shv-02-bom2.facebook.com
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE   VERSION
80/tcp  open  http      proxygen-bolt
443/tcp open  ssl/https
2 services unrecognized despite returning data. If you know the service/version,
 please submit the following fingerprints at https://nmap.org/cgi-bin/submit.cgi
?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port80-TCP:V=7.93%I=7%D=2/29%Time=65E0AA12%P=x86_64-pc-linux-gnu%r(GetR
SF:equest,B3,"HTTP/1\.1\x20301\x20Moved\x20Permanently\r\nLocation:\x20htt
SF:ps:///\r\nContent-Type:\x20text/plain\r\nServer:\x20proxygen-bolt\r\nDa
SF:te:\x20Thu,\x2029\x20Feb\x202024\x2016:00:18\x20GMT\r\nConnection:\x20c
SF:lose\r\nContent-Length:\x200\r\n\r\n")%r(HTTPOptions,B3,"HTTP/1\.1\x203
```

Output Report of Python Script Written By S M RAIHAN KABIR

Vulnerability Report for https://www.facebook.com/

Target Host: www.facebook.com

Security Headers Check:

- **Content-Security-Policy**: default-src data: blob: 'self' https://*.fbsbx.com 'unsafe-inline' *.facebook.com *.fbcdn.net 'unsafe-eval';script-src *.facebook.com *.fbcdn.net *.facebook.net *.google-analytics.com *.google.com 127.0.0.1:* 'unsafe-inline' blob: data: 'self' connect.facebook.net 'unsafe-eval';style-src fonts.googleapis.com *.fbcdn.net data: *.facebook.com 'unsafe-inline';connect-src *.facebook.com facebook.com *.fbcdn.net *.facebook.net wss://*.facebook.com:* wss://*.whatsapp.com:* wss://*.fbcdn.net attachment.fbsbx.com ws://localhost:* blob: *.cdninstagram.com 'self' http://localhost:3103 wss://gateway.facebook.com wss://edge-chat.facebook.com wss://snaptu-d.facebook.com wss://kaios-d.facebook.com/ v.whatsapp.net *.fbsbx.com *.fb.com;font-src data: *.gstatic.com *.facebook.com *.fbcdn.net *.fbsbx.com;img-src *.fbcdn.net *.facebook.com data: https://*.fbsbx.com facebook.com *.cdninstagram.com fbsbx.com fbcdn.net connect.facebook.net *.carriersignal.info blob: android-webview-video-poster: googleads.g.doubleclick.net www.googleadservices.com *.whatsapp.net *.fb.com *.oculuscdn.com *.tenor.co *.tenor.com *.giphy.com;media-src *.cdninstagram.com blob: *.fbcdn.net *.fbsbx.com www.facebook.com *.facebook.com data: *.tenor.co *.tenor.com https://*.giphy.com;frame-src *.doubleclick.net *.google.com *.facebook.com www.googleadservices.com *.fbsbx.com fbsbx.com data: www.instagram.com *.fbcdn.net https://paywithmybank.com/ https://sandbox.paywithmybank.com/;worker-src blob: *.facebook.com data:;block-all-mixed-content;upgrade-insecure-requests;
- **Strict-Transport-Security**: max-age=15552000; preload
- **X-Frame-Options**: DENY
- **X-XSS-Protection**: 0
- **X-Content-Type-Options**: nosniff
- **Cross-Origin-Resource-Policy**: cross-origin
- **Cross-Origin-Opener-Policy**: unsafe-none;report-to="coop_report"
- **Cross-Origin-Embedder-Policy**: MISSING
- **Public-Key-Pins**: MISSING
- **Expect-CT**: MISSING
- **Feature-Policy**: MISSING

- **Feature-Policy**: MISSING
- **Referrer-Policy**: MISSING
- **X-Permitted-Cross-Domain-Policies**: MISSING

Security Header Issues:

- **Cross-Origin-Embedder-Policy** is missing.

Missing Cross-Origin-Embedder-Policy header can make the application vulnerable to Cross-Origin Embedder Policy (COEP) attacks, where an attacker can load the application as an embedder or a nested document in a malicious website.

- **Public-Key-Pins** is missing.

Missing Public-Key-Pins header can expose the application to Man-in-the-Middle (MitM) attacks by allowing attackers to impersonate the server using fraudulent certificates.

- **Expect-CT** is missing.

Missing Expect-CT header can expose the application to Certificate Transparency (CT) policy violations, allowing attackers to use fraudulent certificates without detection.

- **Feature-Policy** is missing.

Missing Feature-Policy header can expose the application to various risks associated with allowing or restricting specific browser features. It helps prevent unauthorized access to features like geolocation, microphone, and camera.

- **Referrer-Policy** is missing.

Missing Referrer-Policy header can leak sensitive information by sending referrer headers to external domains. It helps control how much information is included in the referrer header when navigating to external links.

- **X-Permitted-Cross-Domain-Policies** is missing.

Missing X-Permitted-Cross-Domain-Policies header can expose the application to Cross-Domain Policy (XDP) misconfigurations, allowing unauthorized access to resources from different domains.

Open Ports Check:

- **Port 80**: OPEN

HTTP (Hypertext Transfer Protocol) port is vulnerable to various attacks including cross-site scripting (XSS), SQL injection, cross-site request forgery (CSRF), and DDoS attacks.

- **Port 443**: OPEN

HTTPS (Hypertext Transfer Protocol Secure) port is vulnerable to various attacks including cross-site scripting (XSS), SQL injection, cross-site request forgery (CSRF), and DDoS attacks.

GitHub Link of Python Scripts:

The End