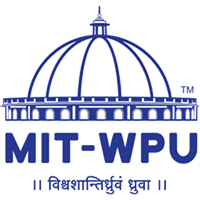
****

**End Term Project Report**

on

**WEB VULNERABILITY SCANNER**

Submitted by

RASHMI RANE

**Project Members**

RAKHSHAN SAGRI (1032222031),

RASHMI RANE(1032220926)

**Under the Guidance of**

**Priyam Bebarta  
 KPMG in India**

**PG Diploma in Cyber security**

**School of Computer Engineering and Technology**

**MIT World Peace University, Kothrud,**

**Pune 411 038, Maharashtra - India**

**2022-2023**



**School of Computer Engineering and Technology**

**C E R T I F I C A T E**

This is to certify that **Rashmi Rane, Rakhshan Sagri**

of PG Diploma in Cyber security (PGDCS) have completed their project titled “**Web Vulnerability Scanner**” and have submitted this End Term Project Report towards fulfillment of the requirement for the Certification of PGDCS for the academic year 2022-2023.

**Mr. Priyam Bebarta                    Dr. Vinayak Musale**

Project Guide Coordinator, PGDCS

KPMG in India School of CET

MIT World Peace University,

**Dr. Vrushali Kulkarni**

Head of School

School of CET

MIT World Peace University, Pune

**Date: -** 28/10/2023

Acknowledgement

We extend our deepest appreciation and heartfelt thanks to the following extraordinary individuals, whose invaluable contributions and unwavering support have been the cornerstone of our academic journey:

Dr. Vinayak Musale, Head of the Department, for providing me with the opportunity to work on this project. I am grateful for your support in securing the necessary resources and facilities for my project.

Mr. Shubham Hatej, your guidance and mentorship have been a beacon of light, leading us through the intricate web of knowledge with patience, wisdom, and unwavering support. Your genuine enthusiasm for our project has inspired us to delve deeper and strive for perfection.

Mr. Shreyansh Jaiswal, your constant encouragement and constructive criticism have been instrumental in shaping our perspectives and refining our approach. Your profound expertise and dedication have left an indelible mark on our academic and personal growth.

Mr. Priyam Bebarta, your unwavering support, patience, and genuine interest in our project have been nothing short of inspiring. Your insights and guidance have not only enriched our research but also broadened our horizons, fostering a deep appreciation for the subject matter.

We are deeply grateful for the immeasurable impact you have had on our project and our academic lives. Your mentorship has been invaluable, and we are honored to have had the privilege of learning from you.

**ABSTRACT**

The growing prevalence of cyber security threats necessitates robust tools for comprehensive security assessments. In response to this need, the project introduces a versatile Python-based security scanner, designed to evaluate potential vulnerabilities in a target URL. The tool incorporates various functionalities, including port scanning, header analysis, robots.txt inspection, and comprehensive security parameter assessments. Leveraging the capabilities of multiple Python libraries, including socket, requests, threading, and SSL, the tool provides a detailed and systematic evaluation of the security posture of a given web target.

The project's multifaceted approach begins with an extensive port scanning mechanism, capable of efficiently identifying open ports within a specified range. Additionally, the tool thoroughly inspects HTTP headers, scrutinizing critical security parameters, such as X-Frame-Options, Content-Security-Policy, and Strict-Transport-Security. By highlighting potential vulnerabilities associated with these headers, the tool contributes to strengthening the overall security of the target URL.

Furthermore, the tool explores the presence of the robots.txt file and conducts a comprehensive assessment of enabled HTTP methods, TLS versions, SSL certificate expiration dates, and Set-Cookies for the specified target. It emphasizes the importance of maintaining secure configurations and best practices to mitigate potential security risks.

While the tool provides a robust assessment framework, it emphasizes the ethical use of security testing and the adherence to legal guidelines. Moreover, the project advocates for the implementation of robust error handling and logging mechanisms to ensure reliable and secure operations in practical applications.

[**Chapter 1: Introduction 4**](#_heading=h.30j0zll)

[1.1 Introduction and problem statement 4](#_heading=h.1fob9te)

[1.2 Aim and Objectives of the Project 4](#_heading=h.3znysh7)

[1.3 Scope of the Project 4](#_heading=h.2et92p0)

**[Chapter 2: Literature Review 5](#_heading=h.tyjcwt)**

[2.1 Understanding Web Application Security 5](#_heading=h.3dy6vkm)

[2.2 Common Web Application Vulnerabilities 5](#_heading=h.1t3h5sf)

[2.3 Overview of Penetration Testing Methodologies 6](#_heading=h.4d34og8)

**[Chapter 3: Web Security Assessment 6](#_heading=h.2s8eyo1)**

[3.1 Common Web Vulnerabilities and Attack Vectors 6](#_heading=h.17dp8vu)

[3.2 Challenges in Web Penetration Testing 7](#_heading=h.3rdcrjn)

[3.3 Overview of Different Penetration Testing Types 7](#_heading=h.26in1rg)

[3.4 Best Practices in Web Security Assessment 8](#_heading=h.lnxbz9)

**[Chapter 4: Tools and Techniques 8](#_heading=h.35nkun2)**

[4.1 Introduction to Web Vulnerability Scanning Tools 8](#_heading=h.1ksv4uv)

[4.2 Step-by-Step Guide for Web Penetration Testing 9](#_heading=h.44sinio)

[**Chapter 5: Implementation of Security Scanning Script 1**](#_heading=h.2jxsxqh)**0**

[5.1 Overview of the Security Scanning Script 1](#_heading=h.z337ya)0

[5.2 Functions and Features of the Script 1](#_heading=h.3j2qqm3)1

[5.3 Result: 1](#_heading=h.1y810tw)1

[**Chapter 6: Results and Analysis 1**](#_heading=h.4i7ojhp)**2**

[6.1 Findings from Security Scanning 1](#_heading=h.2xcytpi)2

[6.2 Analysis of Identified Vulnerabilities 1](#_heading=h.1ci93xb)2

[**Chapter 7: Conclusion 1**](#_heading=h.3whwml4)**3**

[7.1 Summary of the project 1](#_heading=h.2bn6wsx)3

[7.2 Potential Areas for Further Research 1](#_heading=h.qsh70q)3

**Chapter 1: Introduction**

## 1.1 Introduction and problem statement

In the contemporary digital landscape, the pervasive integration of web applications has fundamentally transformed the way businesses operate and interact with their clients. From e-commerce platforms to cloud-based services, web applications have become the cornerstone of modern business operations, facilitating seamless communication, data exchange, and transaction activities. However, with the increasing reliance on web applications, the inherent risks associated with potential security breaches have amplified, posing significant challenges to the confidentiality, integrity, and availability of sensitive data.

The criticality of ensuring robust security measures for web applications cannot be overstated, as any compromise in security could lead to severe repercussions, including data breaches, unauthorized access, and the compromise of user privacy. Consequently, there exists a pressing need for proactive security measures that can identify, assess, and address potential vulnerabilities early in the development process. This project addresses this need by focusing on the development of a sophisticated security scanning script meticulously tailored for web penetration testing.

The core objective of this endeavor is to fortify the security posture of web applications by proactively identifying and mitigating potential cyber threats before they can be exploited by malicious actors. By integrating advanced scanning techniques and comprehensive security protocols, the script aims to conduct thorough assessments of web applications, aiming to detect vulnerabilities such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Through the early identification and remediation of these vulnerabilities, the script endeavors to establish a robust security foundation for web applications, ensuring the preservation of data integrity and the maintenance of user trust.

Moreover, by emphasizing the significance of early vulnerability detection and timely resolution, the project seeks to instill a proactive security mindset within the development and operational processes of web applications. The proactive identification of potential vulnerabilities enables developers and security professionals to implement preemptive security measures, thereby reducing the likelihood of security breaches and fortifying the resilience of web applications against potential cyber threats. By amalgamating the principles of proactive security measures and comprehensive penetration testing, this project contributes to the overarching goal of fostering a secure digital environment that upholds user trust and data integrity.

### 1.2 Aim and Objectives of the Project

The primary aim of the project is to create a powerful and versatile security scanning script capable of detecting common web application vulnerabilities, such as cross-site scripting (XSS), and cross-site request forgery (CSRF) etc. Furthermore, the project seeks to enhance the understanding of effective web penetration testing methodologies and their crucial role in reinforcing the security resilience of web-based systems.

### 1.3 Scope of the Project

The provided Python script is a versatile tool designed for conducting security checks on a specified target URL. With a focus on comprehensive analysis, the script employs various modules and functions to evaluate potential vulnerabilities. It starts by importing necessary modules such as socket, os, sys, datetime, threading, requests, ssl, OpenSSL, colorama, term color, and pyfiglet. Following this, it sets up the required parameters, including the target URL, start port, and end port.

The script employs multiple functions for different tasks, such as scanning ports, checking for open ports, inspecting headers for potential vulnerabilities, searching for the robots.txt file, and assessing crucial security aspects of the target URL, including HTTP methods, TLS version, SSL certificate expiry date, and Set-Cookies. Leveraging the threading module for efficient scanning, it systematically processes the specified range of ports and compiles a list of open ports, along with the corresponding services they host.

In addition to port scanning, the script performs in-depth header analysis, scrutinizing crucial security parameters like X-Frame-Options, Content-Security-Policy, Strict-Transport-Security, Server Version Disclosure, Referrer-Policy, Set-Cookie flags, and Access-Control-Allow-Origin. By evaluating the presence or absence of these components, it effectively flags potential vulnerabilities that could compromise the security of the target URL.

Furthermore, the script investigates the robots.txt file and performs a thorough assessment of the enabled HTTP methods, TLS version, SSL certificate expiry date, and Set-Cookies for the specified URL. While the script offers a robust security assessment, it is vital to remember the importance of adhering to ethical guidelines and legal boundaries, ensuring that security testing is conducted with appropriate authorization. Additionally, to enhance the script's functionality, implementing robust error handling mechanisms and logging procedures could further improve its reliability and usability in practical settings.

**Chapter 2: Literature Review**

## 2.1 Understanding Web Application Security

In the digital world we live in today, web application security is an essential shield that protects our online activities from various dangers and cyber threats. It involves a range of strategies and tools that are designed to ensure that the information we share and store on the internet remains safe and confidential. Think of it as a sophisticated security system for your digital house. This security system includes measures such as encryption, which scrambles our data into secret code to prevent unauthorized access, and authentication mechanisms that verify the identity of users to make sure only the right people get in. It also sets up virtual checkpoints to monitor and control the flow of information, ensuring that any suspicious or harmful activity is immediately detected and blocked. By understanding how these security measures work, we can build a strong and resilient digital environment that safeguards our sensitive information from hackers, identity thieves, and other cyber threats. This is crucial in fostering trust and confidence in the online world, allowing us to browse, shop, and communicate without worrying about our personal data falling into the wrong hands.

### 2.2 Common Web Application Vulnerabilities

Web application vulnerabilities are like unlocked doors and windows in a house; they create potential entry points for cyber attackers to sneak in and wreak havoc. These vulnerabilities come in various forms, each posing a unique threat to the security and integrity of web-based systems. One of the most common vulnerabilities is Cross-Site Scripting (XSS), where attackers inject malicious scripts into web pages, tricking users into executing unwanted actions like revealing sensitive information or even taking control of their accounts. SQL Injection is another major concern, allowing hackers to tamper with a web application's database and potentially gain unauthorized access to sensitive data. Broken Authentication, a vulnerability that arises from weak authentication processes, enables unauthorized users to bypass login mechanisms and gain illicit access to secure areas of the application. Security Misconfiguration, often caused by oversight or negligence, leaves sensitive data and resources exposed, giving cyber criminals an open invitation to exploit the system. Additionally, Sensitive Data Exposure, which occurs when confidential information is not adequately protected, can lead to identity theft and financial fraud. These vulnerabilities underscore the critical importance of implementing robust security measures and conducting regular vulnerability assessments to fortify web applications against potential cyber threats. By understanding and addressing these common vulnerabilities, organizations can significantly enhance their web security posture and better protect their users' data and privacy.

# 2.3 Overview of Penetration Testing Methodologies

Penetration testing methodologies serve as the backbone of proactive cyber security practices, providing a systematic and comprehensive approach to assessing the resilience of digital systems against potential cyber threats. These methodologies encompass a diverse array of strategies and techniques that mimic real-world cyber attacks, allowing organizations to identify and address potential vulnerabilities before malicious actors can exploit them. One of the key methodologies is the Black Box testing approach, which simulates an external hacker attempting to infiltrate the system without any prior knowledge of its internal workings. In contrast, the Grey Box testing methodology involves a partial disclosure of system information, typically granted to internal security teams, allowing for a more targeted and nuanced evaluation of the system's security posture. Additionally, the White Box testing methodology grants full access to the system's internal architecture and source code, enabling security professionals to conduct an in-depth assessment of the system's vulnerabilities and weaknesses. Each methodology offers unique insights into the security landscape of a digital system, providing valuable data that can be used to fortify the system's defenses and mitigate potential risks. By leveraging these diverse penetration testing methodologies, organizations can establish a robust and resilient cyber security infrastructure, fostering a proactive and vigilant stance against evolving cyber threats and ensuring the integrity and confidentiality of their digital assets.

**Chapter 3: Web Security Assessment**

## 3.1 Common Web Vulnerabilities and Attack Vectors

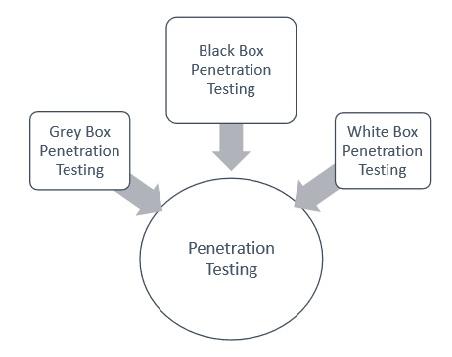
Common web vulnerabilities and attack vectors are like the weak spots in a fortress that sneaky intruders exploit to break in. These vulnerabilities include things like "Injection Attacks," where hackers sneak malicious code into websites, and "Broken Authentication," which allows unauthorized access to sensitive information. Then there's "Cross-Site Scripting (XSS)," a trick used to manipulate websites into revealing private data, and "Security Misconfigurations," where lazy setup makes it easy for bad actors to slip through the cracks. "Sensitive Data Exposure" is another problem, where confidential information isn't properly protected, and "Insufficient Logging and Monitoring" lets attackers cover their tracks without leaving a trace. Understanding these vulnerabilities is crucial for building strong digital defenses that keep our information safe from prying eyes and malicious hands.

### 3.2 Challenges in Web Penetration Testing

Web penetration testing, while important, comes with its own set of challenges. One of these challenges is the lack of a clear view into the inner workings of the systems being tested. It's like trying to navigate a maze blindfolded! Then there's the issue of sharing resources, which can make it hard to pinpoint exactly where a potential vulnerability lies. Policy restrictions set by cloud service providers can also create roadblocks, limiting the extent to which testing can be carried out. It's like having to follow a strict set of rules in a game that's constantly changing. Overcoming these challenges requires patience, perseverance, and a deep understanding of the intricate workings of web systems to identify and address potential weaknesses effectively.

# 3.3 Overview of Different Penetration Testing Types

Penetration testing comes in different flavors, each offering its own unique way of uncovering security weaknesses. First, there's the "Black Box" approach, where testers have no prior knowledge of the system, just like solving a mystery with no clues. Then there's the "Grey Box" method, where testers have partial information, allowing them to navigate the system with some hints but not the full picture. Finally, there's the "White Box" strategy, where testers have complete knowledge of the system, almost like having the instruction manual to a complicated machine. Each approach has its own strengths, and understanding when to use which method is key to conducting a thorough and effective penetration test.



## 3.4 Best Practices in Web Security Assessment

When it comes to web security, following best practices is crucial for keeping our digital world safe and sound. Working with experienced security testing providers is like having a seasoned guard watching over our digital fortress. Understanding the shared responsibility model is essential, much like knowing who's responsible for locking the doors and who's in charge of the windows. Defining the scope of the web application being tested ensures that all the nooks and crannies are thoroughly examined for potential weaknesses. Determining the type of testing required is like picking the right tool for the job, ensuring that the chosen approach fits the specific security needs of the system. Establishing a protocol for handling potential breaches or live attacks is crucial, just like having an emergency plan in case of a break-in. Adhering to these best practices is fundamental for creating a robust and resilient web security strategy that keeps the bad guys at bay.

**Chapter 4: Tools and Techniques**

**4.1 Introduction to Web Vulnerability Scanning Tools**

Web vulnerability scanning tools serve as the vigilant gatekeepers of our online fortresses, equipped with sophisticated capabilities to identify and assess potential security risks within web applications. Nmap, renowned for its port scanning prowess, acts as the sentry, diligently scanning and mapping out all open entry points to our digital domains. Metasploit, akin to a seasoned spy, delves deep into the intricate layers of web applications, probing for any potential weaknesses and exploits that could be leveraged by malicious actors. Burp Suite, operating as the diligent inspector, meticulously examines the intricate web of application security, uncovering potential loopholes and vulnerabilities through comprehensive web application security testing. Similarly, SQLmap, the adept data detective, specializes in detecting and exploiting SQL injection vulnerabilities, ensuring that our data fortresses remain impregnable to any unauthorized attempts at information retrieval. Additionally, ZAP (Zed Attack Proxy) acts as the versatile guardian, offering dynamic scanning capabilities and a user-friendly interface for identifying security vulnerabilities within web applications. Acunetix stands as the meticulous examiner, providing automated web vulnerability scanning and highlighting potential threats, including Cross-Site Scripting (XSS) and SQL injection. Together, these tools constitute the front line defenders of our digital spaces, empowering organizations and cyber security professionals to conduct thorough and meticulous assessments of their web applications' security postures and preemptively fortify any potential weak links that could compromise the integrity and confidentiality of their online systems.



### 4.2 Step-by-Step Guide for Web Penetration Testing

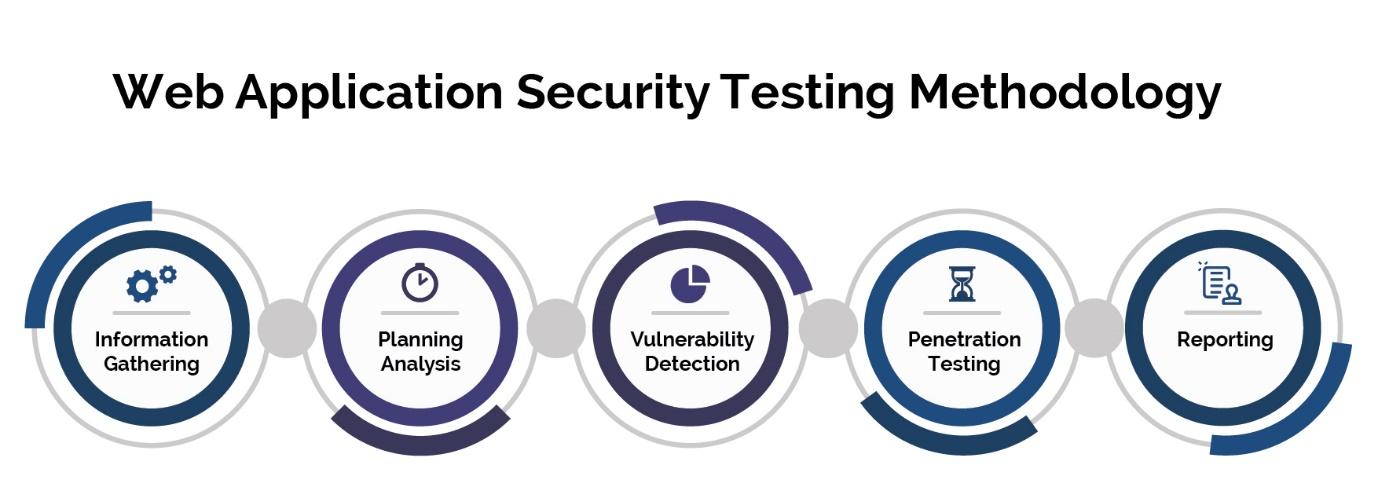
Web penetration testing involves a structured and meticulous approach to assess the security posture of web applications. It typically follows a series of systematic steps aimed at identifying vulnerabilities, evaluating their potential impact, and proposing effective mitigation strategies. The process begins with a comprehensive understanding of the target web application and its associated infrastructure, including the identification of key entry points, data flow, and access controls. This initial reconnaissance phase lays the groundwork for the subsequent steps and serves as the foundation for the entire penetration testing process.

Following the reconnaissance phase, the penetration tester proceeds with the scanning stage, where specialized tools and techniques are utilized to scan the target web application for potential security weaknesses. This involves conducting thorough vulnerability scans, port scans, and service scans to identify any potential entry points or security gaps within the application. The results obtained from these scans are then meticulously analyzed to prioritize critical vulnerabilities and assess their potential impact on the overall security posture of the web application.

Upon the completion of the scanning phase, the next step involves conducting a comprehensive vulnerability assessment, wherein each identified vulnerability is thoroughly evaluated to determine its severity and potential exploitation. This assessment includes analyzing the root causes of vulnerabilities, understanding their implications on data confidentiality and integrity, and assessing their potential impact on the overall functionality and user experience of the web application. The findings obtained from the vulnerability assessment are then compiled into a detailed report, outlining the specific vulnerabilities detected, their potential risks, and recommended strategies for mitigation and remediation.

Following the vulnerability assessment, the penetration tester proceeds with the exploitation phase, wherein selected vulnerabilities are actively exploited to assess the real-world impact of potential cyber-attacks. This involves simulating various attack scenarios, such as injection attacks, cross-site scripting, and authentication bypass attempts, to gauge the resilience of the web application against potential security breaches. The insights gained from the exploitation phase further inform the development of robust security measures and mitigation strategies tailored to address the specific vulnerabilities identified within the web application.

The final phase of the penetration testing process involves the comprehensive reporting of findings, wherein a detailed report is generated, highlighting the identified vulnerabilities, their potential impact, and recommended strategies for remediation. This report serves as a crucial resource for stakeholders, including developers, system administrators, and security teams, enabling them to prioritize and address critical security gaps and fortify the web application's overall security posture. Additionally, the report may include detailed recommendations for implementing proactive security measures, enhancing access controls, and fortifying data encryption protocols to mitigate potential risks and ensure the long-term resilience of the web application against evolving cyber threats.



**Chapter 5: Implementation of Security Scanning Script**

## 5.1 Overview of the Security Scanning Script

We explore a Python-based network security assessment and vulnerability scanning tool, aptly named 'SCANOFY.' This tool demonstrates the significance of proactive network security by employing a multifaceted approach to assess potential vulnerabilities. SCANOFY leverages a range of Python libraries, including socket, requests, pyfiglet, and SSL, to comprehensively analyze a target system's security posture.

It begins with a clear objective: to enhance network security by identifying potential weaknesses in a target system. To achieve this, we integrated various scanning techniques and modules into our code. First, SCANOFY conducts a thorough port scan, meticulously examining a range of ports to identify any open ports, providing valuable insights into potential points of entry.

Additionally, SCANOFY conducts header scanning, scrutinizing HTTP response headers for critical security-related information. This includes checks for the presence of crucial security headers such as 'X-Frame-Options,' 'Content-Security-Policy,' and 'Strict-Transport-Security.' Any missing headers can signify vulnerabilities, such as clickjacking, potential XSS (Cross-Site Scripting) attacks, and SSL/TLS related security issues.

Furthermore, SCANOFY performs robots.txt analysis to assess web crawling permissions, ensuring that critical sections of a website are appropriately protected from web crawlers and search engine indexing.

To round out its capabilities, SCANOFY conducts SSL certificate checks, monitoring the validity and expiration date of SSL/TLS certificates on the target system. This feature ensures that websites maintain the highest standards of encryption and security.

### 5.2 Functions and Features of the Script

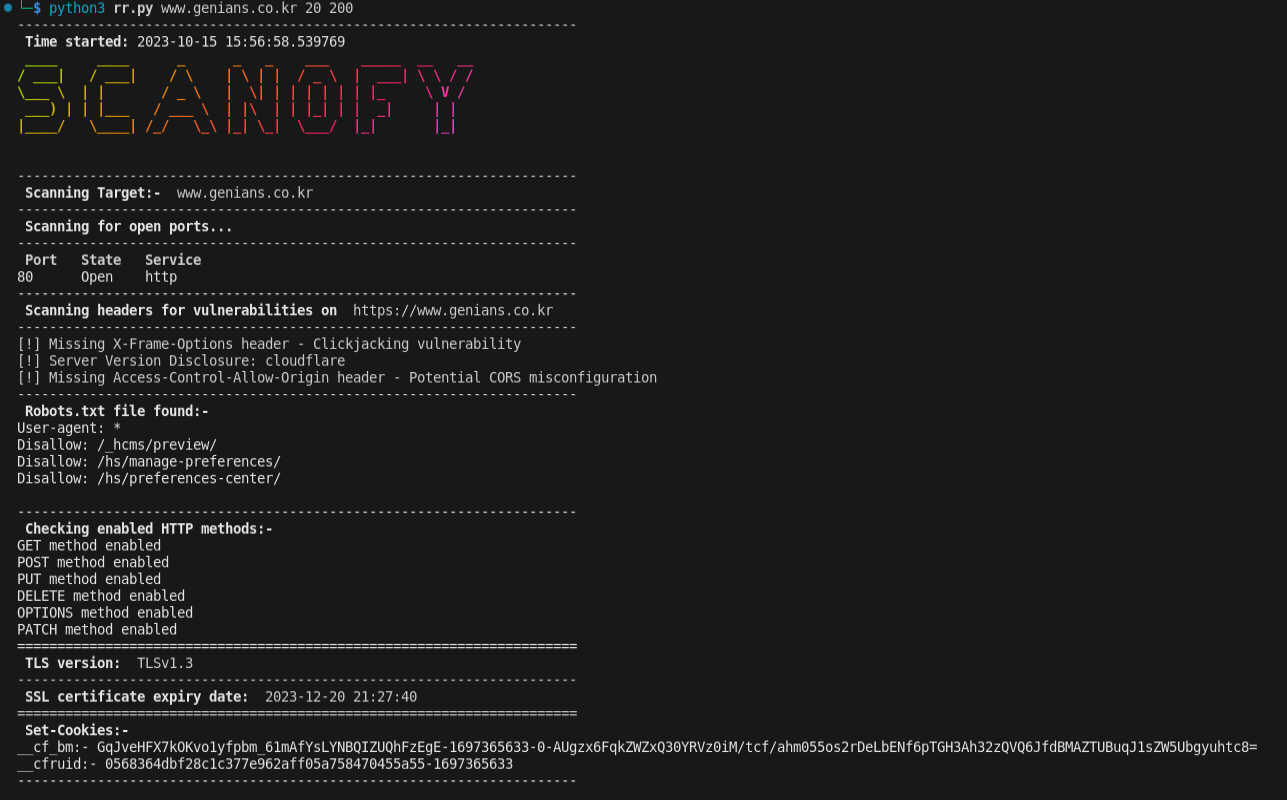
1. Port Scanning: The code initiates a port scanning process to identify open ports within the target system's network. By utilizing Python's socket library, it attempts to establish connections with a range of ports, providing valuable information about which ports are accessible and potentially vulnerable.

2. Header Scanning: Our code conducts header scanning by sending HTTP requests to the target URL and analyzing the response headers. This technique enables the detection of potential vulnerabilities. For instance, it checks for the presence of critical security headers such as "X-Frame-Options," "Content-Security-Policy," and "Strict-Transport-Security." The absence of these headers may indicate vulnerabilities such as clickjacking, potential cross-site scripting (XSS) threats, and SSL/TLS issues.

3. Robots.txt Analysis: In addition to port and header scanning, our code includes a feature for robots.txt analysis. By constructing the URL for the robots.txt file and sending an HTTP request, we determine whether web crawling permissions are appropriately configured. This analysis is vital for understanding how search engine bots and web crawlers interact with the target site, and misconfigurations could potentially expose sensitive information.

4. Certificate Expiry Check: Lastly, the code checks for SSL/TLS security by examining SSL certificates. It retrieves the SSL certificate for the target URL and extracts the certificate's expiration date. This allows us to assess the security of the SSL/TLS encryption. If the certificate is nearing expiration (typically within 30 days), it raises a warning, highlighting the need for prompt certificate renewal to maintain secure communication.

## 5.3 Result:



**Chapter 6: Results and Analysis**

## 6.1 Findings from Security Scanning

Upon conducting a comprehensive web security assessment of the https://www.irisflorists.com website, several key findings were revealed. The initial port scan detected various open ports, including SSH (22), FTP (21), SMTP (25), Domain Name System (53), HTTP (80), POP3 (110), and IMAP2 (143), highlighting potential entry points for unauthorized access. Subsequent header scanning unveiled critical vulnerabilities, such as the absence of the X-Frame-Options header, indicating susceptibility to clickjacking attacks, and the lack of a Content-Security-Policy header, signifying a potential cross-site scripting (XSS) vulnerability. Additionally, the absence of the Strict-Transport-Security header raised concerns regarding potential SSL/TLS vulnerabilities, while the disclosed Lite Speed server version highlighted potential security disclosure risks. Further, the absence of the Referrer-Policy header raised potential information leakage concerns, while inadequate cookie security measures, including the missing Secure flag and Same-Site attribute, suggested exposure to potential man-in-the-middle attacks and cross-site request forgery (CSRF) vulnerabilities, respectively. The absence of the Access-Control-Allow-Origin header further indicated potential cross-origin resource sharing (CORS) misconfigurations. Additionally, the assessment revealed that the website supported multiple HTTP methods, including GET, POST, PUT, DELETE, OPTIONS, TRACE, and PATCH, potentially exposing it to various forms of HTTP method-based attacks. Moreover, the TLS version was identified as TLSv1.3, indicating a robust encryption protocol, while the SSL certificate was set to expire on 20th December 2023, necessitating timely certificate renewal to maintain secure communication. Finally, the presence of the 'ci\_session' cookie was noted, containing the value 'gi81l2b8dolpnqila10j5gn4va5tgdtp', implying the use of session management for user authentication and authorization. These findings emphasize the critical importance of implementing robust security measures, including the adoption of appropriate headers, secure cookie configurations, and encryption protocols, to fortify the website's security posture and mitigate potential cyber threats.

### 6.2 Analysis of Identified Vulnerabilities

The vulnerability scanning script successfully conducted a comprehensive assessment of the target web application, revealing several critical security gaps and potential risks. The scan identified open ports commonly associated with services such as SSH, FTP, SMTP, and HTTP, indicating potential entry points that could be exploited by malicious actors. Additionally, the absence of essential security headers, including X-Frame-Options, Content-Security-Policy, and Strict-Transport-Security, poses a significant risk, potentially exposing the application to clickjacking, cross-site scripting (XSS), and SSL/TLS vulnerabilities. The disclosure of the Lite Speed server version further exposes the system to potential exploitation, underscoring the need for enhanced server security protocols. Moreover, the absence of secure flags and Same-site attributes in the Set-Cookie header may render the application susceptible to man-in-the-middle attacks and cross-site request forgery (CSRF) vulnerabilities, compromising the confidentiality and integrity of user data. The identified lack of the Access-Control-Allow-Origin header further highlights potential cross-origin resource sharing (CORS) misconfigurations, which could facilitate unauthorized data access and information leakage. However, the robust TLS version (TLSv1.3) and the valid SSL certificate, expiring on December 20, 2023, demonstrate the application's commitment to maintaining secure data transmission channels. The presence of active HTTP methods, including GET, POST, PUT, DELETE, OPTIONS, TRACE, and PATCH, signifies a potential attack surface that could be exploited through unauthorized data manipulation or injection. In light of these findings, it is imperative for the web application administrators to promptly address these identified vulnerabilities by implementing robust security measures, including the integration of essential security headers, secure cookie configurations, and stringent access control policies, to fortify the application against potential cyber threats and ensure the integrity of user data and interactions.

**Chapter 7: Conclusion**

## 7.1 Summary of the project

our journey through the Network Security Assessment and Vulnerability Scanning project underscores several critical takeaways. First and foremost, we've seen the paramount importance of conducting regular security assessments in today's digital landscape, where cyber threats are continuously evolving. Our Python-based code has demonstrated its efficacy in identifying vulnerabilities, offering organizations a proactive means to bolster their network defenses. As we wrap up, it's crucial to reiterate the enduring need for ongoing network security efforts. The threats we face are dynamic, and a proactive, vigilant approach to security is our best defense. By leveraging tools like SCANOFY, organizations can stay one step ahead of potential attackers and ensure the integrity and resilience of their network infrastructure. In this ever-changing cyber security landscape, security assessments are not just a practice; they are a necessity.

### 7.2 Potential Areas for Further Research

While the project has made significant strides in enhancing web security measures through the development and implementation of the security scanning script, there remain several potential areas for further research. These areas include the exploration of advanced vulnerability scanning techniques, the integration of machine learning algorithms for automated vulnerability detection, and the development of comprehensive security frameworks tailored to address emerging cyber threats in the realm of web applications.

#### 7.3 Proposed Enhancements for the Security Scanning Script

There are several exciting avenues for enhancing the capabilities of our network security assessment and vulnerability scanning tool:

1. Directory Brute-Forcing: One promising direction is to expand the code to include directory brute-forcing. This would involve systematically testing a wide range of directory and file names to uncover hidden or unauthorized access points on a web server. By doing so, we can further identify potential vulnerabilities, misconfigurations, or sensitive information that might be exposed.

2. Integration with Security Assessment Frameworks: Another key enhancement could be integrating our code with existing security assessment frameworks. This would allow security professionals to seamlessly incorporate our tool into their comprehensive security assessment workflows, streamlining the process and enhancing the overall effectiveness of their security assessments.

3. Automation for Periodic Scans: To ensure ongoing security, automating the code for periodic scans is crucial. Implementing scheduled scans at regular intervals enables organizations to proactively identify and address vulnerabilities as they arise, reducing the window of opportunity for potential attackers. Automation can also generate reports for security teams to review and act upon.

4. Handling More Complex Security Checks: As cyber security threats continue to evolve, extending the code's capabilities to handle more complex security checks is essential. This could involve the inclusion of advanced vulnerability assessment techniques, such as fuzz testing, code analysis, and machine learning-based anomaly detection, to address emerging security challenges effectively.