A Security Assessment Tool and Generating Reports

Mr. Dr. Om Prakash Samantray[1], M. Lahar Krishna[2], P. Priyanka[3], M. Rambabu[4], J. Bhavani[5]

[#1]Associate Professor in Department of CSE, Raghu Engineering College, Visakhapatnam.

[#2 #3 #4 #5] B. Tech with Specialization of Computer Science and Engineering – Cyber Security in Raghu Institute of Technology, Visakhapatnam.

ABSTRACT:

The project is about developing a custom security assessment tool and reporting system for analyzing the vulnerabilities of existing websites. This project aims to bridge the gap by creating a tailored tool for specific website needs and generate concise, actionable reports. We define the website elements and vulnerabilities to be assessed (e.g., content security policy, refer policy, cookies, etc.). Implementing a risk-based scoring system to prioritize critical vulnerabilities for immediate attention. The outcome of the project is that it can detect the vulnerabilities present on the website and generate a detailed report by providing the reasons for the vulnerabilities.

1. INTRODUCTION:

VulneraBot is a state-of-the-art solution meticulously crafted to equip website owners with the necessary capabilities to detect and rectify fundamental vulnerabilities present in their platforms. This innovative tool streamlines the entire process by performing comprehensive scans of websites, thereby furnishing users with lucid and comprehensive assessments of their site's security posture.

By leveraging advanced technological advancements, VulneraBot intelligently examines every nook and cranny of websites, leaving no stone unturned in its pursuit of identifying potential security weaknesses. Through its user-friendly interface and seamless functionality, this intuitive bot seamlessly guides website owners, regardless of their technical expertise, to effortlessly comprehend and mitigate any vulnerabilities affecting their sites.

The comprehensive scans carried out by VulneraBot delve into various aspects of a website's security, including but not limited to, detecting susceptible entry points, unpatched software, weak passwords, and insecure configurations. Armed with the knowledge gained from these exhaustive scans, website owners can promptly take action to bolster their site's security infrastructure and safeguard against potential cyber threats.

Moreover, VulneraBot goes beyond mere identification and diagnosis of vulnerabilities by providing users with actionable insights and recommendations for

remediation. This invaluable feature aids website owners in prioritizing and addressing each vulnerability efficiently, enabling them to fortify their site's defenses systematically. By empowering users with clear and concise insights into their site's security posture, VulneraBot ensures that website owners can make informed decisions to mitigate risks effectively.

After the scanning process is completed, VulneraBot provides users with a comprehensive report that provides valuable insights into the security of their website. This report includes a categorization of the website's security grade into four distinct levels: A (Excellent), B (Good), C (Average), and D (Poor). Each grade is associated with a specific score and color, allowing users to easily and efficiently assess the security status of their website.

Additionally, VulneraBot goes beyond just grading the security of the website. It also evaluates the website's performance against 11 vulnerability tests, highlighting the number of tests that have been successfully passed. This provides website owners with a deeper understanding of their website's overall security posture and helps them identify potential vulnerabilities that need to be addressed.

We understand that security is of paramount importance and assure our users that the installation of VulneraBot does not pose any risks to their website. Stringent security measures have been put in place to protect user data, ensuring the safeguarding of all sensitive information. As website owners, you can rest assured that your developer websites remain highly secure, with passwords encrypted to further enhance the protection of sensitive information.

By utilizing VulneraBot, website owners can confidently evaluate their website's security grade, identify vulnerabilities, and take proactive steps to enhance their security measures. Our aim is to empower website owners with the knowledge and tools they need to maintain a secure online presence and protect their valuable data.

In the ever-evolving landscape of cybersecurity, VulneraBot stands as a reliable and indispensable tool for website owners, offering a proactive approach to ensure their platforms remain impervious to attacks. With its cutting-edge technology, ease of use, and robust security assessments, VulneraBot sets a new standard in enabling website owners to effortlessly bolster their site's security and protect valuable digital assets.

2. LITERATURE SURVEY:

F. Wu et. al. has studied that, Vulnerability detection is an import issue in information system security. They propose the deep learning method for vulnerability detection. They presented three deep learning models, namely, convolution neural network (CNN), long short term memory (LSTM) and convolution neural network - long short term memory (CNN-LSTM) [1].

Kessel et. al. has studied that core principle of open science is the clear, concise and accessible publication of empirical data, including “raw” observational data as well as processed results [2].

Nguyen et. al. has studied that organizations and developers are underestimating security issues on their system. They proposed a protective and extensible solution for automatically detecting both the Web application vulnerabilities and malicious Web shells. Based on the original THAPS, they proposed E-THAPS that has a new detecting mechanism, improved SQLi, XSS and vulnerable functions detecting capabilities [3].

Cruz et. al. has studied that software applications continue to become more complex and attractive to cyber-attackers, enhancing resilience against cyber threats became essential. They aimed to provide more robust solutions, different approaches for vulnerability detection in different stages of the application life-cycle [4].

Alazmi et. al. depicted that web applications become increasingly popular for offering data and services among businesses and organizations they also become more susceptible to security risks. Many organizations rely today on Web Vulnerability Scanners (WVSs) to identify vulnerabilities in their web applications. He also stated that, one of the most prevalent types of web application vulnerabilities, SQL Injections (SQLi), can often go undetected by WVSs. OWASP ZAP is an open-source web vulnerability scanner that allows security professionals to develop rules to improve vulnerability detection capability [5].

3. PROPOSED SYSTEMS:

The way a vulnerable bot works is depicted in figure 1. The process starts with downloading the bot using jQuery commands. Once downloaded, the bot checks if a token is available in session storage. If a token is found, the bot proceeds to validate the token. The bot prompts the user for their credentials, including a username and password.

Once the user enters their username, the bot verifies the validity of the username using AJAX. Similarly, it also validates the entered password. After successfully validating the credentials, if they are determined to be true, the bot adds the token to session storage. However, if the credentials are found to be false, the bot gives the user the option to try again.

If the user chooses to try again, they can enter their credentials once more, and the bot will again validate them using AJAX. Once the token is successfully stored in session storage, the bot proceeds to download itself onto the system. It then proceeds to validate the website and generate reports.

Furthermore, the bot displays the generated report to the user. Additionally, it provides the option for the user to perform the scan again. If the user chooses not to perform the scan again, the bot exits.

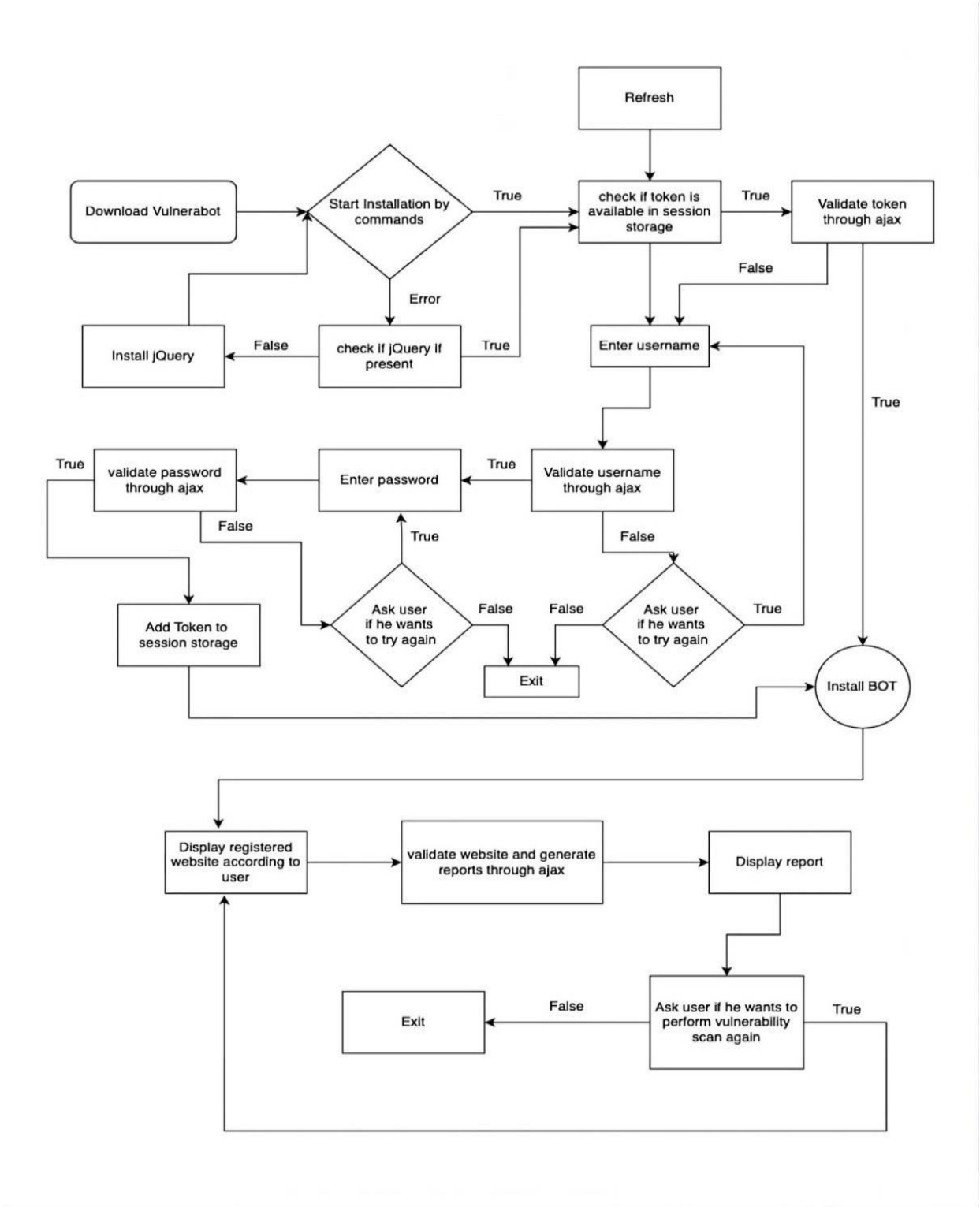


Fig. (1) Flow chart how a vulnerabot works

4. IMPLEMENTATION:

We utilized two prominent websites, Inavap and Drupal, to conduct a comprehensive vulnerability assessment. These websites were securely hosted on the reliable Netlify hosting server.

To enhance our vulnerability detection capabilities, we implemented vulneraBot by integrating jQuery Commands into the website console. To fully leverage the features of this tool, it is imperative to register the host website. This registration enables us to execute various tasks related to the website's security.

Upon registration, vulnerabot prompts for a Username and Password. If the provided Username matches the entries in the database, it displays a pop-up indicating that the user has been found and requests the Password for further authentication. Conversely, if the Username or Password does not match, the bot displays an error message stating "Wrong Password" and offers the option to retry by pressing "OK".

The Username and password functionality is an essential aspect of our security measures. By employing this feature, we can verify the existence of a user based on the provided userId. It yields a true value if the user is located confirming their authenticity, and false otherwise.

We prioritize the implementation of robust security measures, such as vulnerability scanning and user authentication, to ensure the utmost protection for our hosted websites.

The password function is an essential component of the system, as it ensures secure access for users. When a password is entered, it is transmitted to the server where it is compared with the encrypted password stored in the database. If the comparison returns true, indicating a successful match, the password, along with a token, is transmitted to the website. The token is then stored as a session token within the website.

It’s crucial to acknowledge that with every new login, the session token undergoes a change. This means that once a user closes their browser tab or session, the session token expires, enhancing security.

By employing this password function and the use of session tokens, the system ensures that only authenticated users can access the website, thereby safeguarding sensitive data and protecting user privacy.

After clicking the start button, the application initiates an API call to the website server and proceeds with conducting a vulnerability scan. The scan is performed using various Python Libraries, which meticulously checks for any potential vulnerabilities. Once the scan is completed, a comprehensive report highlighting the identified vulnerabilities is generated. To facilitate easy viewing and distribution, this report is then converted into a PDF format by utilizing the html2PDF library.

Token Generation algorithm:

```
FUNCTION generateRandomToken(searchRow):
character='0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'token = ''
FOR i FROM 0 TO 11:
    randomIndex = RANDOM_INT(0, LENGTH(characters) - 1)
    token += characters[randomIndex]
users = GET_ALL_USERS() // Get all users from the database
getRow = FIND_ROW_WITH_SEARCH_ROW(users, searchRow)  // Getting the row from database
rowIndex = INDEX_OF(getRow) + 2  // Adding 2 because the index starts from 0 and there's a header row
tokenColumnIndex = 2 // Assuming the token column is the third column (index 2)
SET_TOKEN_IN_DATABASE(rowIndex, tokenColumnIndex, token)
RETURN token
```

The password is encrypted by using SHA 256 and the below Algorithm :

```
FUNCTION hashPasswordWithSaltAndIterations(password, salt, iterations):
  hashedPassword = SHA256(password + salt)
  //SHA256 is the encryption method to secure passwords
  FOR i FROM 1 TO iterations:
    hashedPassword = SHA256(hashedPassword + salt)
  RETURN HEX_ENCODE (hashedPassword)
  //HEX_ENCODE will encode the password.
```

5. RESULTS:

Vulnerabot, an advanced security tool, offers an in-depth analysis of website security performance, providing invaluable insights to protect against potential vulnerabilities. Once the scanning process begins, the bot diligently examines various aspects of the website's security infrastructure. Within minutes, it generates a comprehensive grade, ranging from A to F, accurately reflecting the website's level of security. Additionally, Vulnerabot furnishes an informative score out of 100, coupled with a detailed count of

Table 1. Tabular form of Results

Test	Drupal	Inavap
Content Security Policy	Fail	Fail
Contribute	Pass	Pass
Cookies	Fail	Pass
Cross origin Resource Policy	Pass	Pass
Redirection	Pass	Pass
Referrer policy	Pass	Pass
Strict transport Security	Pass	Fail
Sub resource Integrity	Pass	Pass
X Content Type Options	Pass	Fail
X Frame Options	Pass	Fail
X Xss Protection	Pass	Pass

the number of test cases successfully passed during the evaluation.

As a testament to its commitment to transparency and accessibility, Vulnerabot offers users the convenience of downloading a comprehensive report. This report encapsulates all the crucial details uncovered during the scanning process, empowering website owners and administrators to take effective measures in fortifying their online presence. By leveraging the information within this report, individuals can proactively address any identified security gaps and bolster their website's resilience against potential threats. The test results is shown in Table 1.

This comprehensive report of two websites are shown in Fig.(2). It provides an in-depth analysis of the vulnerabilities discovered on the Drupal and Inavap websites. The purpose of this assessment was to identify any weaknesses present in these sites and to determine the reasons behind their failure to meet the specified test cases.

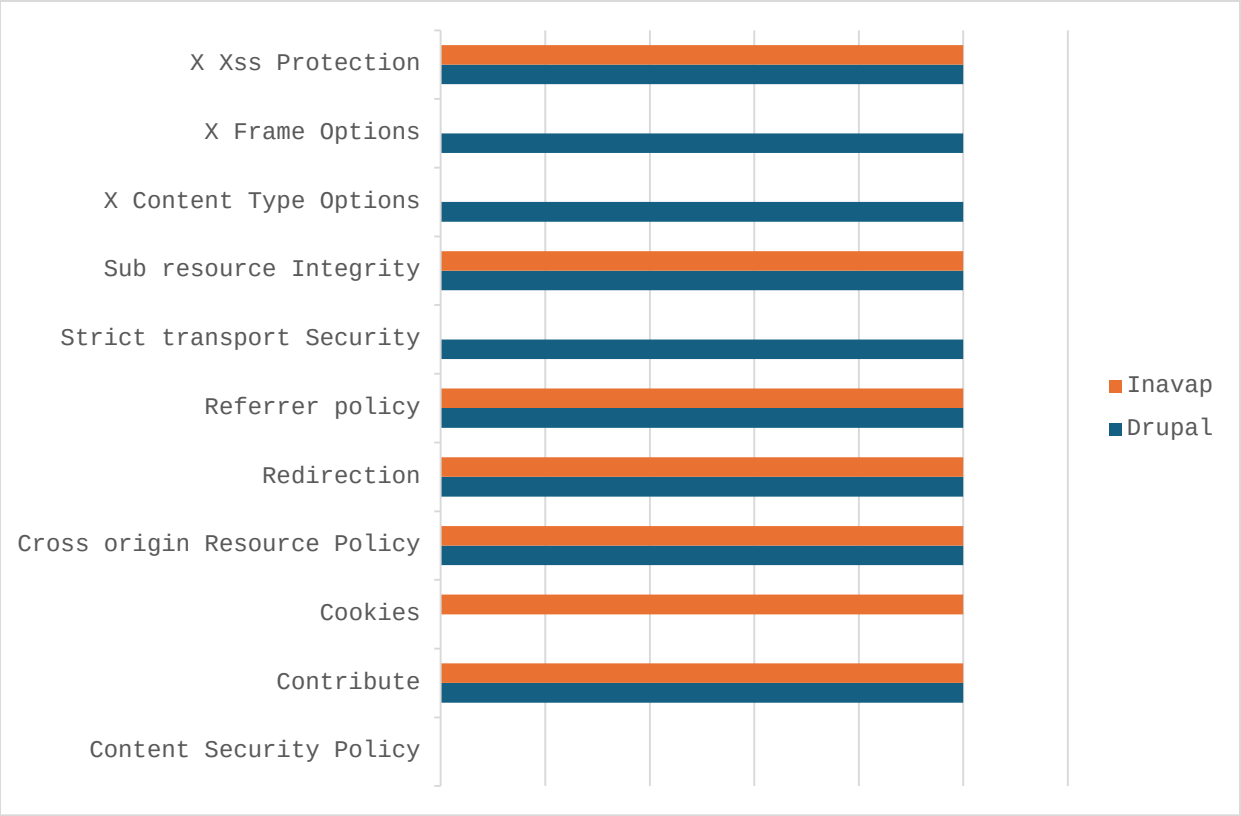
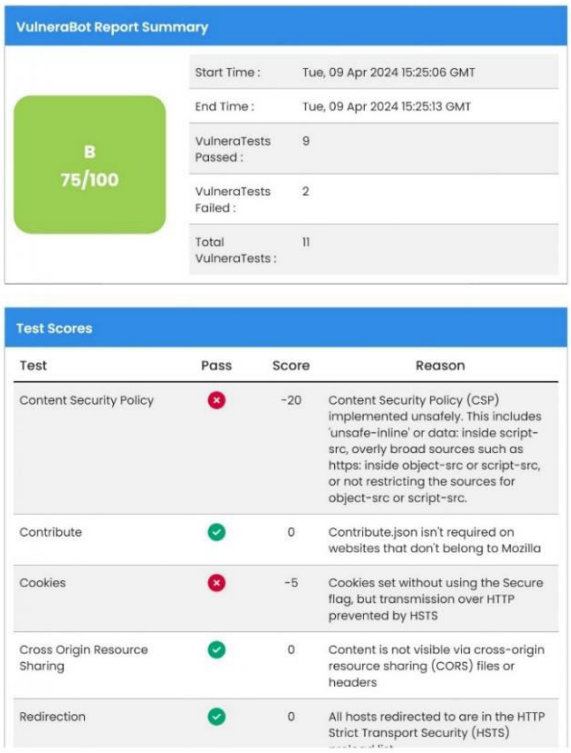


Fig.(2)Graphical Representation

Drupal Website:

Upon reviewing both reports, it is evident that there are notable disparities between the vulnerabilities found on the Drupal website and those discovered on the Inavap website. The Drupal website achieved a commendable grade B and attained a score of 75 out of 100. It successfully passed 9 out of 11 test cases, indicating a relatively secure online platform.

However, there are areas for improvement in terms of fortifying its security, particularly with regard to addressing vulnerabilities such as Content Security Policy and cookies. The Website results are shown in Fig.(3) and Fig(4).



VulneraBot Report Summary

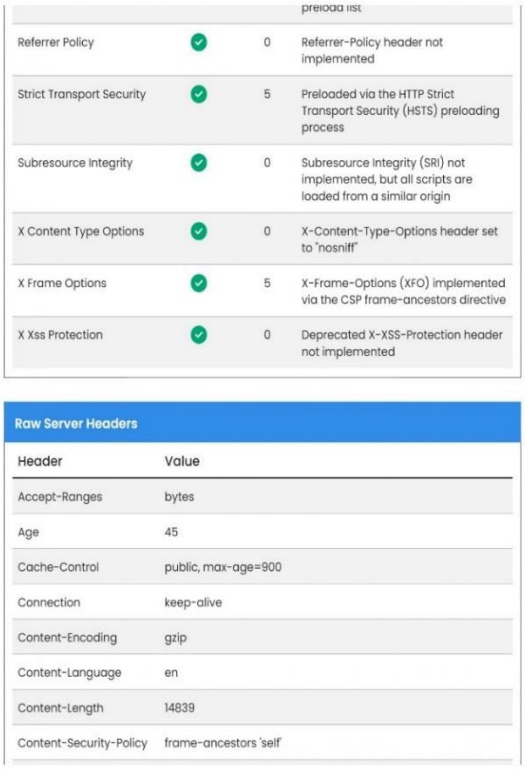
B
75/100

Start Time :	Tue, 09 Apr 2024 15:25:06 GMT
End Time :	Tue, 09 Apr 2024 15:25:13 GMT
VulneraTests Passed :	9
VulneraTests Failed :	2
Total VulneraTests :	11

Test Scores

Test	Pass	Score	Reason
Content Security Policy	✗	-20	Content Security Policy (CSP) implemented unsafely. This includes 'unsafe-inline' or data: inside script-src, overly broad sources such as https: inside object-src or script-src, or not restricting the sources for object-src or script-src.
Contribute	✓	0	Contribute.json isn't required on websites that don't belong to Mozilla
Cookies	✗	-5	Cookies set without using the Secure flag, but transmission over HTTP prevented by HSTS
Cross Origin Resource Sharing	✓	0	Content is not visible via cross-origin resource sharing (CORS) files or headers
Redirection	✓	0	All hosts redirected to are in the HTTP Strict Transport Security (HSTS)

Fig.(3) Drupal Report



			preload list
Referrer Policy	✓	0	Referrer-Policy header not implemented
Strict Transport Security	✓	5	Preloaded via the HTTP Strict Transport Security (HSTS) preloading process
Subresource Integrity	✓	0	Subresource Integrity (SRI) not implemented, but all scripts are loaded from a similar origin
X Content Type Options	✓	0	X-Content-Type-Options header set to "nosniff"
X Frame Options	✓	5	X-Frame-Options (XFO) implemented via the CSP frame-ancestors directive
X Xss Protection	✓	0	Deprecated X-XSS-Protection header not implemented

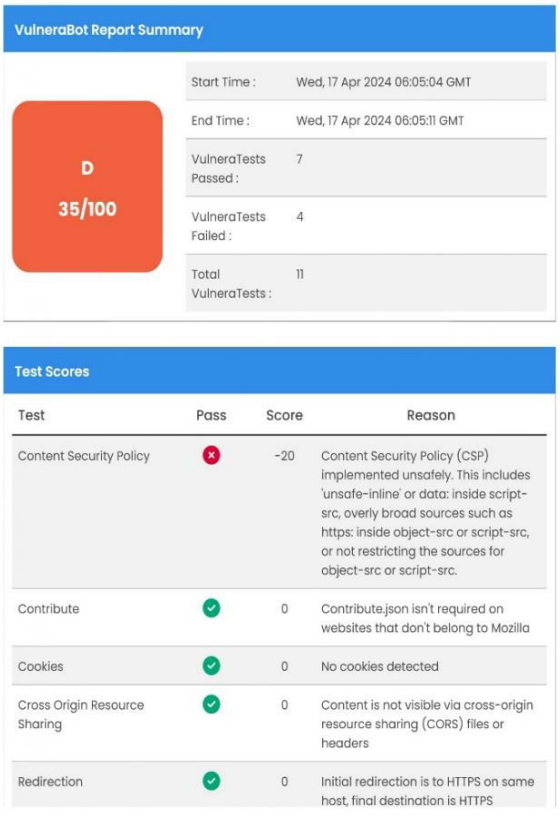
Raw Server Headers

Header	Value
Accept-Ranges	bytes
Age	45
Cache-Control	public, max-age=900
Connection	keep-alive
Content-Encoding	gzip
Content-Language	en
Content-Length	14839
Content-Security-Policy	frame-ancestors 'self'

Fig.(4) Drupal Report

Inavap Website:

Upon close examination of the inavap reports, it is evident that the website in question currently holds a grade D with a score of 35 out of 100. Additionally, it successfully passed 7 out of 11 test cases conducted. It is important to note that this website does not meet the desired level of security, necessitating a focused effort on addressing crucial areas such as content security policy, X-frame option, strict transport security, and more. Taking proactive measures to strengthen these security aspects will be imperative in order to ensure the safety and protection of user information. The results are shown in Fig.(5) and Fig.(9).



VulneraBot Report Summary

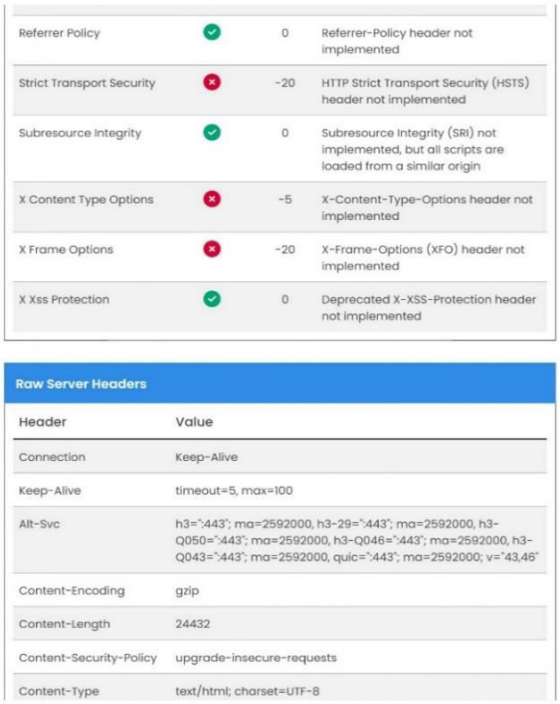
D
35/100

Start Time :	Wed, 17 Apr 2024 06:05:04 GMT
End Time :	Wed, 17 Apr 2024 06:05:11 GMT
VulneraTests Passed :	7
VulneraTests Failed :	4
Total VulneraTests :	11

Test Scores

Test	Pass	Score	Reason
Content Security Policy	✗	-20	Content Security Policy (CSP) implemented unsafely. This includes 'unsafe-inline' or data: inside script-src, overly broad sources such as https: inside object-src or script-src, or not restricting the sources for object-src or script-src.
Contribute	✓	0	Contribute.json isn't required on websites that don't belong to Mozilla
Cookies	✓	0	No cookies detected
Cross Origin Resource Sharing	✓	0	Content is not visible via cross-origin resource sharing (CORS) files or headers
Redirection	✓	0	Initial redirection is to HTTPS on same host, final destination is HTTPS

Fig.(5) INAVAP Report



Referrer Policy	✓	0	Referrer-Policy header not implemented
Strict Transport Security	✗	-20	HTTP Strict Transport Security (HSTS) header not implemented
Subresource Integrity	✓	0	Subresource Integrity (SRI) not implemented, but all scripts are loaded from a similar origin
X Content Type Options	✗	-5	X-Content-Type-Options header not implemented
X Frame Options	✗	-20	X-Frame-Options (XFO) header not implemented
X Xss Protection	✓	0	Deprecated X-XSS-Protection header not implemented

Raw Server Headers

Header	Value
Connection	Keep-Alive
Keep-Alive	timeout=5, max=100
Alt-Svc	h3=":443"; ma=2592000, h3-29=":443"; ma=2592000, h3-Q050=":443"; ma=2592000, h3-Q046=":443"; ma=2592000, h3-Q043=":443"; ma=2592000, quic=":443"; ma=2592000; v="43,46"
Content-Encoding	gzip
Content-Length	24432
Content-Security-Policy	upgrade-insecure-requests
Content-Type	text/html; charset=UTF-8

Fig.(6) INAVAP Report

6. CONCLUSION:

In order to ensure the security of websites, it is crucial to have a thorough understanding of potential vulnerabilities. To aid in this process, a bot has been developed to scan websites and provide comprehensive reports on their performance. These reports not only highlight existing vulnerabilities but also offer insights into the reasons behind them, enabling users to take necessary measures to safeguard their websites.

However, it must be acknowledged that existing scanners often have a limited scope, focusing solely on a single vulnerability while disregarding other critical issues. To overcome this limitation, this paper proposes the introduction of a web vulnerability bot that performs an extensive scan of websites, addressing major vulnerabilities and ensuring their security. This advanced bot generates detailed reports encompassing all vulnerabilities discovered, as well as raw headers.

Moreover, through rigorous experimental testing, our scanner has proven to be exceptionally effective, showcasing its potential for practical implementation. Thus, it stands as a reliable and viable solution for organizations and individuals seeking to enhance the security of their websites.

Further Work:

Vulnerability Identification and Suggestions:- Vulnerabot is capable of identifying vulnerabilities in software systems.- Once vulnerabilities are detected, the bot will provide suggestions to the user on how to fix or mitigate those vulnerabilities.

Automated Solutions:- In addition to providing suggestions, Vulnerabot can also offer automated solutions to address certain vulnerabilities.- The bot will have the capability to automatically implement fixes for vulnerabilities that it is trained to handle.

Vulnerability Remediation:- Vulnerabot can directly resolve some vulnerabilities, without requiring manual intervention from the user.- The bot will be trained to identify and fix specific types of vulnerabilities that it is capable of addressing programmatically.

Scope of Vulnerabilities Handled:- The text indicates that Vulnerabot can access and fix the vulnerabilities that it has been trained to handle.- This suggests that the bot's capabilities are limited to a specific set of known vulnerabilities, and it may not be able to address all possible vulnerabilities that may exist in a system.

7. REFERENCES:

[1]. F. Wu, J. Wang, J. Liu and W. Wang, "Vulnerability detection with deep learning," *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, Chengdu, China, 2017, pp. 1298-1302, doi: 10.1109/CompComm.2017.8322752.

[2]. Kessel, M., & Atkinson, C. (2024). Promoting open science in test-driven software experiments. *Journal of Systems and Software*, *212*, 111971.

[3]. Nguyen, V.G, Le, H. T., Lu, D. N., & Nguyen, N. H. (2016). A solution for automatically malicious web shell and web application vulnerability detection. In *Computational Collective Intelligence: 8th International Conference, ICCCI 2016, Halkidiki, Greece, September 28-30, 2016. Proceedings, Part I 8* (pp. 367-378). Springer International Publishing.

[4]. Cruz, D. B., Almeida, J. R., & Oliveira, J. L. (2023). Open Source Solutions for Vulnerability Assessment: A Comparative Analysis. *IEEE Access*.

[5]. Alazmi, S., & de Leon, D. C. (2023, May). Customizing OWASP ZAP: A Proven Method for Detecting SQL Injection Vulnerabilities. In *2023 IEEE 9th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)* (pp. 102-106). IEEE.