Smart Approach for Vulnerability Detection Using React

Dr. Dimple Chawla, Assistant Professor, VIPS-TC, GGSIP University

Ms. Swati Jain, Assistant Professor, VIPS-TC, GGSIP University

Abstract

In the realm of web security, a comprehensive vulnerability detection and prevention system serves as the cornerstone of a robust and holistic security strategy. Within this framework, a myriad of tools and practices converge, each offering unique strengths and capabilities to mitigate vulnerabilities effectively. By integrating a diverse array of tools and approaches, organizations can fortify their defences and proactively safeguard against potential threats in the ever-evolving landscape of cyber security. This research paper delves into a comprehensive examination of various vulnerability scanning tools, spotlighting modern approaches such as OWASP ZAP, Nessus, OpenVAS, and Nikto, alongside Web Application Firewalls like ModSecurity, SonarQube, Burp Suite, and Splunk. Furthermore, it proceeds to illustrate an experimental implementation utilizing React for front-end interface. Users input the URL of the web application, enabling the system to conduct vulnerability assessments and subsequently recommend preventive measures based on the detected vulnerabilities. This amalgamation of advanced tools and innovative implementation techniques underscores a proactive approach to web security, emphasizing both detection and prevention in the ongoing battle against cyber threats.

.Keywords: ***Vulnerability Management, Vulnerability Detection, Cross-Site Scripting, SQL Injection, CSRF (Cross-Site Request Forgery), CVSS (Common Vulnerability Scoring System), National Vulnerability Database (NVD), Common Vulnerability Exposures (CVEs), Web Application Firewall (WAF)***

1. Introduction

The vulnerability management is a structured process and comprehensive strategy for identifying, assessing, prioritizing, and addressing security vulnerabilities across an organization's systems, networks, and applications [1]. This process typically comprises four main stages defined by [2] i.e. Identification, Reporting, Remediation and Re-assessment. *Identification* stage uses the vulnerability scanners, manual examination, and asset discovery techniques, vulnerabilities within various systems (including servers, desktops, laptops, mobile devices, and IoT devices) are identified and documented. Each vulnerability is evaluated and assigned scores based on different criteria. In *reporting stage*, identified vulnerabilities are documented and communicated to the relevant stakeholders through a reporting mechanism. This ensures that the appropriate parties are aware of the security issues and can take necessary actions to resolve them. The *remediation stage* involves addressing and resolving the identified security vulnerabilities. This is typically carried out by system owners or technical personnel through various methods such as applying security patches, reconfiguring systems, or implementing additional security controls. Following the *remediation efforts*, a re-assessment is conducted to verify that the security vulnerabilities have been successfully mitigated. This may involve conducting additional scans or manual checks to ensure that the identified vulnerabilities are no longer present [2].

Vulnerability detection is a critical process in cyber security, aimed at identifying and reporting security flaws in software to mitigate the risk and impact of cyber-attacks [3]. It plays a vital role in reducing the attack surface, which refers to the potential security vulnerabilities a malicious actor could exploit to harm an organization [**4**]. A sophisticated vulnerability detection system is designed to fortify online platforms against potential threats in the evolving digital landscape [5]. By employing advanced scanning techniques, such a system can locate common security vulnerabilities like Cross-Site Scripting (XSS), SQL Injection, and Cross-Site Request Forgery (CSRF). Its user-friendly interface and commitment to continuous monitoring make it an invaluable asset for developers and administrators seeking to enhance the security posture of their web applications. This process is crucial in safeguarding against viruses and threats that could compromise system integrity by injecting malicious code to exploit user data and privacy [6]. Through advanced scanning techniques and penetration testing, vulnerability detection tools meticulously analyse web applications or websites for vulnerabilities, and insecure authentication mechanisms. Furthermore, through informative reporting and actionable recommendations, these state-of-the-art systems not only alert users to potential risks but also empower them with the knowledge and tools to proactively prevent exploitation by identified vulnerabilities [7].

With the evolving threat landscape and increasingly sophisticated attack vectors, investing in robust vulnerability detection measures is paramount for staying one step ahead of cyber threats and ensuring the resilience of web-based platforms and services. By proactively identifying and addressing different vulnerabilities, organizations can strengthen their online security posture and mitigate the risk of data breaches, financial loss, and reputational damage [8]. Additionally, vulnerability detection empowers businesses to stay compliant with industry regulations and standards, demonstrating their commitment to safeguarding sensitive information and maintaining the trust of their customers.

2. Related Study

The report provides an overview of significant vulnerabilities that emerged in various systems and technologies over the course of 2019. The identified vulnerabilities had far-reaching implications across industries, impacting cyber security, data privacy, and technological infrastructure. The report highlights the importance of proactive measures to mitigate risks associated with vulnerabilities and emphasizes the need for collaboration among stakeholders to address emerging threats effectively [**9**].

The Common Vulnerability Scoring System (CVSS) is a public framework used to rate the severity and characteristics of security vulnerabilities in information systems. It assigns a numerical score ranging from 0 to 10 to indicate the severity of a vulnerability, with 10 being the most severe. CVSS is vendor-neutral, meaning it can be used to score vulnerabilities across a wide range of software products, including operating systems, databases, and web applications. This consistency allows organizations to assess and prioritize vulnerabilities consistently, regardless of the software vendor [**10, 11**].

The evolution of the Common Vulnerability Scoring System (CVSS) from version 2.0 to version 4.0, highlighting key updates and improvements made with each release. Released in 2007, CVSS v2 was considered a significant improvement over the original version. It addressed inconsistencies present in the earlier version, provided additional granularity, and more accurately reflected the properties of IT vulnerabilities. CVSS 3.0, introduced in June 2015, brought about scoring changes that better reflected real-world vulnerabilities encountered in practice. Changes included adjustments to factors such as the privileges required for successful exploitation and the impact on the attacker if the vulnerability is successfully utilized. CVSS version 3.1, released in June 2019, focused on clarifying and improving the standard. Unlike previous versions, it did not introduce new metrics or metric values, nor did it make major changes to the formulas [9, 10].

The most recent version, CVSS 4.0, was released on Nov. 1, 2023, and introduced various improvements [10]. While specific details about the improvements are not provided, it can be inferred that CVSS 4.0 builds upon the foundation laid by previous versions to enhance the accuracy and effectiveness of vulnerability scoring [11, 12].

The National Vulnerability Database (NVD), from the National Institute of Standards and Technology has disclosed the vulnerability counts. The NVD recorded 28,831 vulnerabilities in 2023, up from 25,081 in 2022 [11]. The data mentioned in [12] indicates a significant increase in the number of newly discovered common IT security vulnerabilities and exposures (CVEs) over the years, with the highest reported annual figure recorded in 2023. In the first week of 2024, internet users worldwide discovered 612 new common IT security vulnerabilities and exposures (CVEs) [10]. This suggests that cyber security threats continue to evolve, requiring vigilance and proactive measures to mitigate risks. The highest reported annual figure for newly discovered CVEs was recorded in 2023, with over 29 thousand vulnerabilities identified shown in Figure 1. This indicates a significant increase in the volume of cyber security vulnerabilities compared to previous years [11, 12].

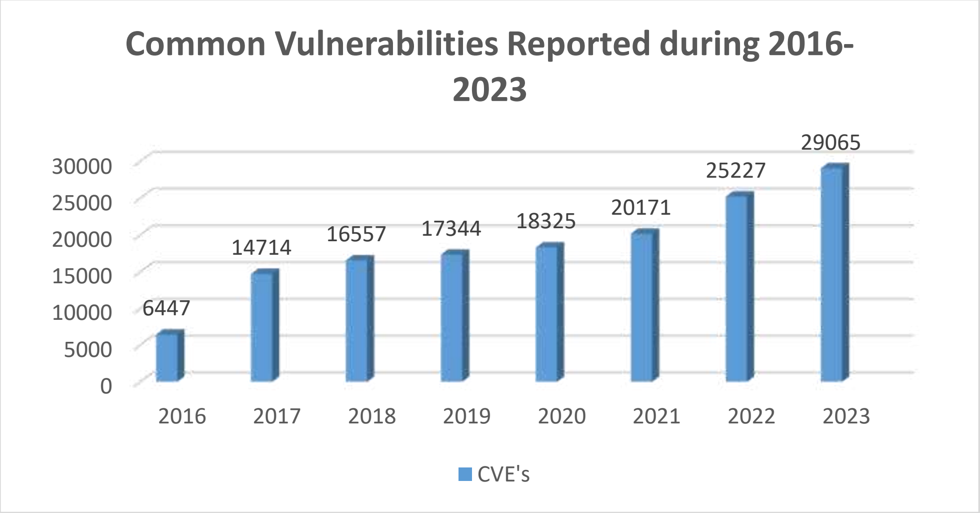


Figure 1: Common Vulnerabilities Reported in years [Vulnerabilities Year-in-Review].

The analysis of vulnerability trends of year 2022 and 2023 shown in Table 1, reveals notable shifts in the types of vulnerabilities reported, highlighting the dynamic nature of cyber security threats and the need for adaptive security measures [11].

Table 1: Common Vulnerability Trends [11]

S. No.	Common Vulnerability Types	Occurrences reported in year 2023	Occurrence compared with year 2022
1.	Cross-Site Scripting (XSS)	5,297	It remained as the most common vulnerability type in both 2022 and 2023, indicating its persistent prevalence as a cyber-security threat.
2.	Structured Query Language Injection (SQLi)	2,261	It also maintained their positions among the top vulnerability types, albeit with varying occurrence rates.
3.	Cross-Site Request Forgery (CSRF)	1,324	It emerged as a new entrant to the top five list in 2023, surpassing other vulnerability types such as improper input validation flaws.
4.	Out-of-Bounds Write	2,066	The improper input validation flaws, which ranked fifth in 2022, did not make the top five list in 2023.
5.	Out-of-Bounds Read	1,068	

3. Analysis of Vulnerability Scanning Tools

In the development of an advanced vulnerability detection and prevention system, thorough requirement analysis is paramount. This entails a meticulous examination of existing systems, if any, to grasp their utilization of hardware, software, network infrastructure, and human resources for managing data-related tasks such as user interactions, resource uploads, and downloads. The objective is to map out the current information system workflow comprehensively, encompassing crucial activities like input, processing, output, storage, and control. By scrutinizing existing processes, we can pinpoint shortcomings and areas for enhancement, ensuring that the forthcoming system effectively addresses web application security needs while adhering to user-friendly design principles and industry standards. Though it may not be feasible to eradicate all vulnerabilities within a specified network, actively identifying and mitigating them where feasible can significantly reduce the likelihood of initial network compromise and impede malicious activities post-compromise.

i. **Different Vulnerability Scanning Tools:** Focusing tools like OWASP ZAP, Nmap, or Nessus indeed offers a more comprehensive approach to vulnerability scanning shown in Table 2. These tools often provide extensive databases of known vulnerabilities, along with advanced scanning techniques to identify potential weaknesses in systems and networks.

Table 2: Common Vulnerability Scanning Tools

	OWASP ZAP	**Nessus**	**OpenVAS**	**Nikto**
Type	OWASP ZAP (Zed Attack Proxy) is an open-source web application security scanner.	It is a proprietary vulnerability scanner developed by Tenable.	Open Vulnerability Assessment System is an open-source vulnerability scanner.	It is an open-source web server scanner designed to identify potential vulnerabilities and misconfigurations.
Scope	Focuses on web application security testing, including identifying common vulnerabilities such as SQL injection, cross-site scripting (XSS), and insecure configurations.	Offers comprehensive vulnerability scanning for networks, operating systems, applications, and cloud environments.	Provides vulnerability scanning and management capabilities for networks, including host discovery, port scanning, and vulnerability detection.	Specializes in web server scanning to detect potential security issues, including outdated software versions, misconfigurations, and known vulnerabilities.
License	Open-source software licensed under the Apache License 2.0.	Proprietary software with commercial and free versions available.	Open-source software licensed under the GNU General Public License (GPL).	Open-source software licensed under the GNU General Public License (GPL).
Features	Includes features such as passive and active scanning, automated spidering, authentication support, and scripting for custom tests.	Offers vulnerability scanning, compliance checks, configuration auditing, asset discovery, and integration with third-party tools.	Provides vulnerability scanning, asset discovery, configuration assessment, reporting, and integration with other security tools.	Offers web server scanning, SSL/TLS scanning, multiple target scanning, custom plugin support, and reporting capabilities.
Usage	User-friendly interface with both GUI and command-line options. Suitable for both beginners and experienced users.	Intuitive interface with customizable scan templates and reporting options. Suitable for both small businesses and enterprise users.	Command-line interface (CLI) with a web-based management interface (Greenbone Security Assistant). May require some configuration and setup.	Command-line tool with straightforward usage. Suitable for experienced users familiar with web server security testing.
Community and Support:	Supported by a large community of security professionals and frequent updates from the OWASP project.	Offers comprehensive technical support, documentation, and user forums for both free and paid versions.	Supported by the community with active development and regular updates. Community support available through forums and mailing lists.	Community-driven project with active development and support available through forums and online resources.
Cost	Free and open-source.	Commercial versions available with pricing based on the number of IP addresses scanned. Limited free version also available.	Free and open-source.	Free and open-source.

When selecting a vulnerability scanning tool, consider on specific requirements, such as the type of assets required to scan, the level of expertise available, budget constraints, and integration with existing security infrastructure. Each tool has its strengths and weaknesses, so it's essential to evaluate them based of organization's needs and priorities. The vulnerability assessment involves an active examination of a network to identify any hosts, software and configurations with vulnerabilities that have not been remediated. It may include unpatched software, network protocols using outdated encryption and security standards, or exposed ports and network services not adequately protected behind firewall. The assessment is based on its strengths for passive tracking and software updating for highlighting software updates. Also detects security vulnerabilities from software and network configurations or inadequate network segmentation. The weakness assessment based on the full insight view from the impact of exploitation or the level of risk. [How to identify cyber security vulnerabilities]

Keeping all software updated is indeed one of the most effective ways to mitigate security vulnerabilities. Vendors regularly release updates to patch security flaws and improve overall system stability. Monitoring vendor advisories and applying patches promptly helps to safeguard against known *vulnerabilities. Passive vulnerability* tracking, which involves monitoring vendor updates and security advisories, is crucial for identifying potential weaknesses in outdated systems. However, the complexity of modern computer networks means that unique vulnerabilities may exist, necessitating more proactive measures. Whereas *Active vulnerability assessment*, often conducted through penetration testing, allows organizations to directly interact with systems and assess their security posture. Penetration testing encompasses various techniques to simulate real-world attacks and identify potential vulnerabilities that may not be apparent through passive tracking alone [3].

By combining passive vulnerability tracking with active measures like penetration testing, organizations can comprehensively identify and mitigate security vulnerabilities, thereby strengthening their overall cyber security posture.

ii. **Types of vulnerabilities assessments**: Several types are given in [2] are as follows:
   a. Host assessment: The host assessment is critical, which may be vulnerable to attacks if not adequately tested or not generated from a tested machine.
   b. Network and wireless assessment: This assessment initiates the practice to prevent unauthorized access to private or public networks and network accessible resources.
   c. Database assessment: The assessment of databases or system for vulnerabilities identifying rogue database or insecure test environment, classifying the sensitive data across an organization’s infrastructure.
   d. Application scans: The identifying of security vulnerabilities in web applications and their source code by automated scans on the front-end or static/ dynamic analysis of source code

The process of scanning initiates with the *first, vulnerability identification* in which penetration testing is performed then to *analysis* for identify the source and root cause of the each vulnerability identified. *Third, risk assessment* is to be performed by prioritizing the vulnerabilities in order to rank the severity score of each vulnerability. The multiple factors is associated like system are affected, what data is at risk, which business functions are at risk, severity of an attack and potential damage as a result of the vulnerability. *Fourth as last is remediation*, this step is for determining the most effective path for mitigation of each vulnerabilities. Remediation steps includes introduction of new security procedures, measures and tools. The updating of operational or configuration changes to development and implementation of a vulnerability patch [2].

4. State of the art system design methodology

The system designed with the purpose of defining many underlining features like as follows:

i. **User Interface and Input Management:** The platform offers a user-friendly web interface powered by React, enabling seamless interaction. Users input the URL of the target web application they want to evaluate for vulnerabilities. The system efficiently manages form submissions and state updates, storing crucial information such as the URL, assessment outcomes, loading status, and duration of the vulnerability assessment process.

ii. **Request Handling and Processing:** Utilizing Axios for HTTP requests, our system efficiently manages communication with the backend. When a user submits a URL through the web interface, Axios sends an HTTP GET request to a dedicated endpoint (/vulnerability) on the backend, built with FastAPI. This request includes the user-provided URL as a parameter. Upon receiving the request, the backend extracts the URL parameter and triggers the vulnerability assessment process tailored to the specified URL.

iii. **Vulnerability Assessment:** Within the backend, the assess_vulnerabilities function conducts a comprehensive evaluation process. It begins by verifying the URL's protocol, prioritizing HTTPS connections while seamlessly transitioning to HTTP if necessary. Using BeautifulSoup, the system parses the HTML content of the webpage, scrutinizing it for recognizable patterns indicative of common vulnerabilities such as XSS, CSRF, and SQL Injection. Additionally, regular expressions are employed to detect specific vulnerabilities based on common indicators within the webpage's content. The assessment further ensures secure connections by checking for HTTPS usage and the presence of SSL certificates. Optionally, specialized scanning functions like scan_sql_injection are employed for targeted detection of vulnerabilities such as SQL Injection.

iv. **Vulnerability Detection and Reporting:** Identified vulnerabilities are consolidated into a comprehensive list for easy reference. In cases where no common vulnerabilities are detected, a clear message is generated to signify the absence of any identified vulnerabilities.

v. **Result Presentation and Navigation System:** The frontend interface showcases the assessment outcomes, presenting a summary that includes any identified vulnerabilities, the assessed URL, and the duration of the assessment process. For detected vulnerabilities, our system provides actionable recommendations. Users can easily navigate to detailed solutions or mitigation suggestions by utilizing a dedicated component, such as VulSolution, enhancing the usability of the platform.

vi. **Security and Cross-Origin Requests:** To facilitate seamless interaction between the React frontend and the backend, the backend is equipped with CORS middleware. This configuration ensures that requests from the frontend are permitted, thus maintaining compliance with the same-origin policy and enhancing the overall security of the web application.

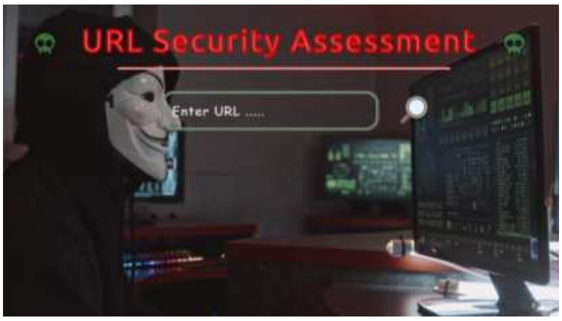
vii. **Continuous Monitoring and Updates:** Beyond the outlined functionalities, vulnerability detection system is dedicated to ongoing vigilance and improvement. This entails routine scanning of registered URLs to promptly detect any new vulnerabilities. Additionally, our commitment extends to updating our vulnerability detection capabilities to effectively combat emerging threats, ensuring the sustained security of our platform and its users.

The system's usability is paramount, accommodating users with diverse technical backgrounds and ensuring accessibility for non-technical personnel. It achieves this through intuitive interfaces and clear navigation paths, making it easily comprehensible and operable for all users. Every element of the system is interactive, boasting a visually captivating graphical user interface (GUI) to enhance engagement. Design principles prioritize variety, avoiding monotony to deliver a delightful user experience that encourages exploration. Moreover, the system upholds high standards of reliability and efficiency in data processing to meet user expectations. Leveraging FastAPI and React, chosen technologies facilitate rapid response times and a seamlessly smooth user journey.

To maintain data integrity, access to system data must be tightly controlled, allowing only authorized personnel to retrieve and modify information. Robust encryption measures are essential to protect user credentials and sensitive data from unauthorized access. User-provided data should be kept confidential, with users retaining exclusive rights to modify their information. Access controls and permission systems play a crucial role in safeguarding against unauthorized data modifications, ensuring that data remains secure and accessible only to those with proper authorization.

5. Stimulation Design

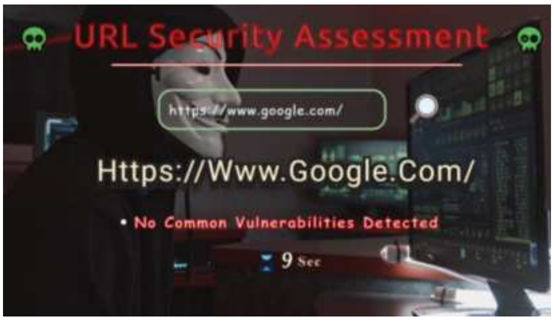
The system is engineered to streamline information flow and management, prioritizing accurate data processing and storage. Leveraging FastAPI's asynchronous capabilities, it ensures efficient handling of data transactions and system operations. User-facing content is designed to be self-explanatory and easily comprehensible, complemented by clear data visualization and reporting techniques that offer users actionable insights into vulnerability detection and prevention. Scalability is a core aspect of the system's design, accommodating increased user loads and data volumes without sacrificing performance. Containerization using Docker and deployment on cloud platforms further enhance scalability, ensuring seamless adaptation to evolving needs. The comprehensive solution for identifying and addressing security vulnerabilities in web applications, homepage of the URL security assessment tool depicted along with Figures 2, 3 and 4. Figure 2 defines the URL security assessment homepage. Figure 3 reports no common vulnerabilities were detected in the assessment of the provided URL. Whereas Figure 4 highlights the vulnerability detected only in 14 seconds, so a link is generated to redirect on a web page for prevention measures of that vulnerability. After clicking on “Prevention Measures”, link will redirect the users to a dedicated webpage containing information and measures to prevent the "Broken Authentication" vulnerability as shown in Figure 5 as example



URL Security Assessment

Enter URL .....

Figure 2: Homepage of URL Security Assessment



URL Security Assessment

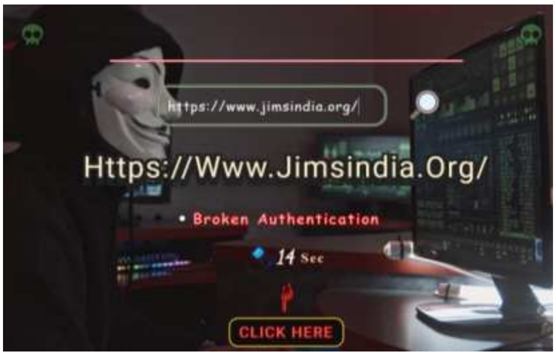
https://www.google.com/

Https://Www.Google.Com/

• No Common Vulnerabilities Detected

9 Sec

Figure 3: No Common Vulnerabilities Detected



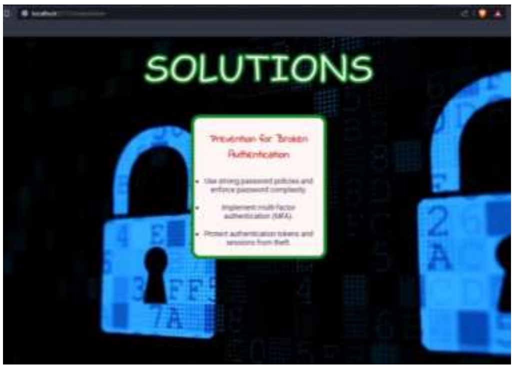
https://www.jimsindia.org/

Https://Www.Jimsindia.Org/

• Broken Authentication

14 Sec

Figure 4: Detected Common Vulnerabilities



SOLUTIONS

Prevention for Broken Authentication

- Use strong password policies and enforce password complexity.
- Implement multi-factor authentication (MFA).
- Protect authentication tokens and sessions from theft.

Figure 5: Preventive Solution

6. Prevention Measures for Vulnerabilities:

Preventive measures for detecting vulnerabilities in web applications are crucial to maintaining robust cyber security. By employing regular security audits and assessments helps in identify potential weaknesses in the application codebase, while adherence to secure coding practices ensures resilient code that mitigates common vulnerabilities like injection attacks and cross-site scripting(XSS), where few of them are listed in Table 3. Utilizing it in the frameworks and libraries with built-in security features, along with implementing strict input validation and data sanitization, further fortifies the application against malicious exploits.

Table 3: Preventive Action & it description

S.No.	Title	Description
i.	"XSS"	"1": "Use proper output encoding.” "2": "Use security libraries like OWASP ESAPI to sanitize input.” "3": "Implement content security policies (CSP)."
ii.	"Command Injection"	"1": "Avoid using user-controlled input in system commands.” "2": "Use proper input validation and sanitization.” "3": "Implement access controls to restrict command execution."
iii.	"Insecure Password Storage"	"1": "Hash and salt passwords before storage.” "2": "Use strong password policies.” "3": "Protect password databases with proper access controls."
iv.	"CSRF"	"1": "Use anti-CSRF tokens in forms.” "2": "Implement SameSite cookie attribute.” "3": "Verify the origin and referrer headers in requests."
v.	"IDOR"	"1": "Implement proper access controls and authorization checks.” "2": "Use indirect references instead of direct object references.” "3": "Validate and sanitize user input used for object references."
vi.	"Sensitive Data Exposure"	"1": "Encrypt sensitive data at rest and in transit.” "2": "Use proper access controls to restrict data access.” "3": "Implement strong authentication and authorization mechanisms."
vii.	"Security Misconfiguration"	"1": "Regularly audit and review server and application configurations.” "2": "Follow security best practices and guidelines.” "3": "Remove unnecessary services and components.”
viii.	"Broken Authentication"	"1": "Use strong password policies and enforce password complexity.” "2": "Implement multi-factor authentication (MFA).” "3": "Protect authentication tokens and sessions from theft."
ix.	"Insecure Deserialization"	"1": "Avoid deserializing data from untrusted sources.” "2": "Implement proper input validation and sanitization.” "3": "Use security libraries for safe deserialization."
x.	"Missing Rate Limiting"	"1": "Implement rate limiting for APIs and sensitive operations.” "2": "Configure request rate limits based on use cases.” "3": "Monitor and log rate-limiting violations."
xi.	"Missing HTTP Security Headers"	"1": "Implement security headers like X-Frame-Options, X-XSS-Protection, etc.” "2": "Set appropriate content security policies (CSP).” "3": "Regularly audit and validate security headers configurations."
xii.	"SQL Injection"	"1": "Use parameterized queries or prepared statements.” "2": "Implement input validation and sanitization.” "3": "Escaping user input used in SQL queries."

Additionally, deploying web application firewalls (WAFs) and employing secure authentication mechanisms and access controls aid in detecting and preventing unauthorized access and malicious activities. By integrating these measures into the development lifecycle and maintaining vigilance through continuous monitoring, organizations can proactively identify and address vulnerabilities, enhancing the overall security posture of their web applications.

7. Conclusion

The organizations can effectively manage their security vulnerabilities and reduce the risk of potential breaches by adopting a systematic approach to security. This approach aims to create a user-friendly, secure, and efficient vulnerability detection and prevention system aligned with system objectives. Regular testing, user feedback, and iterative development should continually refine these requirements. The dynamic nature of cyber security threats, highlighted by vulnerabilities observed in 2019, underscores the importance of proactive risk management. Staying vigilant, implementing effective security measures, and fostering collaboration among stakeholders strengthens resilience against emerging threats. Despite ongoing challenges, statistics emphasize the importance of robust vulnerability management practices, including timely patching, threat intelligence sharing, and proactive risk mitigation. This comprehensive approach helps organizations maintain a strong security posture across their IT infrastructure, safeguarding their digital assets effectively.

8. References

[1] Guoyu Luo, Research on Network Security Vulnerability Detection Method Based on Artificial Intelligence", Journal of Physics: Conference Series, Volume 1651, The 2020 second International Conference on Artificial Intelligence Technologies and Application (ICAITA) 2020 21-23 August 2020, Dalian, China

[2] Imperva a Thales company, "Vulnerability Assessment" https://www.imperva.com/learn/application-security/vulnerability-assessment/#:~:text=A%20vulnerability%20assessment%20is%20a,mitigation%2C%20if%20and%20whenever%20needed.]

[3] Gupta, A., & Sharma, L. S. (2020). Detecting attacks in high-speed networks: Issues and solutions. Information Security Journal: A Global Perspective, 29(2), 51–61. https://doi.org/10.1080/19393555.2020.1722296

[4] Nikolai Mansourov, Djenana Campara," Chapter 6 - Knowledge of vulnerabilities as an element of cybersecurity argument, Editor(s): Nikolai Mansourov, Djenana Campara, In The MK/OMG Press, System Assurance, Morgan Kaufmann, 2011, Pages 147-170, ISBN 9780123814142, https://doi.org/10.1016/B978-0-12-381414-2.00006-3.

[5] An, J.H., Wang, Z. & Joe, I. A CNN-based automatic vulnerability detection. J Wireless Com Network 2023, 41 (2023). https://doi.org/10.1186/s13638-023-02255-2

[6] Nikolai Mansourov, Djenana Campara, Chapter 3 - How to build confidence, Editor(s): Nikolai Mansourov, Djenana Campara, In The MK/OMG Press, System Assurance, Morgan Kaufmann, 2011, Pages 49-80, ISBN 9780123814142, https://doi.org/10.1016/B978-0-12-381414-2.00003-8.

[7] Daniel Albrecht, How to identify cybersecurity vulnerabilities, Cybersecurity education, from the experts, Jan 29, 2024, https://fieldeffect.com/blog/how-to-identify-cybersecurity-vulnerabilities#:~:text=Testers%20use%20a%20combination%20of,vulnerabilities%20hidden%20from%20surface%20assessments.

[8] L. Ma, "Research on Vulnerability Exploitation and Detection Technology Based on Big Data Analysis," 2021 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI), Harbin, China, 2021, pp. 429-435, doi: 10.1109/IAAI54625.2021.9699917.

[9] Dima Bekerman, Sarit Yerushalmi, “The State of Vulnerabilities in 2019” Imperva a thales company, Jan 23, 2020 https://www.imperva.com/blog/the-state-of-vulnerabilities-in-2019/#:~:text=2019%20vulnerabilities%20statistics&text=We%20can%20see%20that%20the,compared%20to%202017%20(14%2C086).

[10] Kinza Yasar, Alexander S. Gillis and Madelyn Bacon, “Common Vulnerability Scoring System (CVSS)”, TechTarget Network, https://www.techtarget.com/searchsecurity/definition/CVSS-Common-Vulnerability-Scoring-System

[11] Vulnerabilities Year-in-Review: 2023, March 27, 2024, https://intel471.com/blog/vulnerabilities-year-in-review-2023

[12] Ani Petrosyan, Common IT vulnerabilities and exposures worldwide 2009-2024, Jan 9, 2024 https://www.statista.com/statistics/500755/worldwide-common-vulnerabilities-and-exposures/