International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 05 | May - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Web based Vulnerabilities Modulation: A Comprehensive Study on Web Vulnerabilities

Raju Ranjan

Department of Computer Science & Engineering, Apex Institute of Technology, Chandigarh University Punjab, India.

Rishabh Sharma

Department of Computer Science & Engineering, Apex Institute of Technology, Chandigarh University Punjab, India.

Anurag Mishra

Department of Computer Science & Engineering, Apex Institute of Technology, Chandigarh University Punjab, India.

Abstract: *In today's digital landscape, where online activities like banking, shopping, and social networking dominate, the security of sensitive data and assets is paramount. This responsibility primarily lies with software and web developers, tasked with safeguarding against prevalent vulnerabilities such as SQL injection, XSS (Cross-Site Scripting), Broken Access Control, Command Execution, CSRF (Cross- Site Request Forgery), among others. This research project delves into the world of web application security, aiming to provide a comprehensive understanding of secure development and deployment practices. By simulating and analyzing these vulnerabilities across different modules, the study sheds light on how attackers can exploit these weaknesses to breach data confidentiality, integrity, and availability, thus violating the CIA triad.*

Keywords: web, vulnerability, XSS, CSFR, SQL injection, Access Control.

I. INTRODUCTION

Web applications serve as the primary gateways for conducting business in the modern cyber era. These applications, developed using a myriad of technologies and programming languages, represent the forefront of digital interactions and transactions. While state-of-the- art software technologies integrate security frameworks within the Software Development Life Cycle (SDLC), addressing the vast spectrum of online security remains a formidable challenge.

The complexity of web applications often introduces vulnerabilities that may evade traditional secure frameworks. These vulnerabilities encompass not only technical flaws but also loopholes in business logic, posing significant risks to data confidentiality, integrity, and availability. Among the most prevalent vulnerabilities are SQL injection, CSRF (Cross-Site Request Forgery), XSS (Cross-Site Scripting), and broken access control.

SQL injection vulnerabilities arise when malicious actors exploit inadequately sanitized inputs to execute unauthorized SQL commands, potentially compromising the entire database. CSRF vulnerabilities enable attackers to manipulate authenticated user sessions and perform unauthorized actions, leading to data breaches or account takeovers. XSS vulnerabilities allow attackers to inject malicious scripts into web pages viewed by other users, facilitating phishing attacks or data theft. Broken access control vulnerabilities occur when inadequate access restrictions enable unauthorized users to access privileged resources or manipulate data.

To confront these challenges, the software development life cycle includes a crucial phase: the web application vulnerability assessment and penetration testing. Within this phase, two primary methodologies emerge: manual web application software vulnerability assessment and automated software vulnerability assessment. These methodologies leverage tools to identify common vulnerabilities that may have eluded detection during the development phase. However, despite robust frameworks and diligent assessment phases, open security research and responsible disclosure initiatives underscore the persistent susceptibility of web applications to evolving threats.

Once a web application goes live, it becomes a focal point for cyber security researchers diligently scrutinizing for vulnerabilities. Their efforts often lead to the discovery of security weaknesses in production environments, prompting appropriate mitigation actions through formal channels. However, the digital landscape also harbors adversaries – hackers who actively seek out web application

vulnerabilities for personal, financial, or political gains. Exploiting identified loopholes in web applications, hackers navigate in a clandestine manner, leaving minimal traces in server logs. This dichotomy underscores the ongoing battle in web application security. While companies and businesses establish clear rules and policies for handling and reporting vulnerabilities, the failure to adhere to responsible disclosure practices can lead to legal and ethical repercussions.

In essence, the introduction of web application vulnerabilities encapsulates a dynamic interplay between technological advancements, security challenges, and human factors. Understanding this landscape is pivotal for navigating the complexities of modern digital ecosystems and safeguarding against evolving threats. This introduction provides a more detailed insight into specific types of web vulnerabilities, enhancing the understanding of their impact and the ongoing efforts to mitigate associated risks in today's digital environment.
The remaining sections are arranged as follows: A few similar works are described in ***Section 2***. ***Section 3*** explains the proposed image encryption technique. The numerical experiments in ***Section 4*** are presented along with typical simulation results. In the final portion, the conclusion is provided.

II. Common Web-Based Vulnerabilities

A. SQL Injection

When a web application's input fields are manipulated, malicious SQL commands can be executed directly on the database by attackers using a serious security issue called SQL injection. Bypassing authentication, gaining access to confidential information, and possibly taking over the database, they can achieve this. SQL injection seriously jeopardises data integrity and confidentiality by taking advantage of the trust that exists between the database and the app. Adopting strategies like parameterized queries, access constraints, and input validation is essential to combating it.
Using a web application login form as a target, an attacker might inject malicious data such as:
login: admin OR '1'='1'

Enterany password here.

The SQL query that is produced could resemble this:

WHERE password = "anypassword" AND username = "admin" OR "1"="1'--"
This instance circumvents the password check since the injected 'OR '1'='1'--' section always evaluates to true. The remainder of the query is commented out by the '--' to ensure that there are no errors. Hence, by using the incorrect password to log in as the admin user, the attacker can access the system without authorization.

B. Cross-Site Scripting (XSS)

Attackers utilise the devious technique known as Cross- Site Scripting (XSS) when user input is handled improperly by web applications. Sometimes they use HTML, CSS, or Flash, but most commonly they smuggle malicious programmes into these applications written in JavaScript. Unbeknownst to you, your browser launches the hidden script when you visit a compromised website.

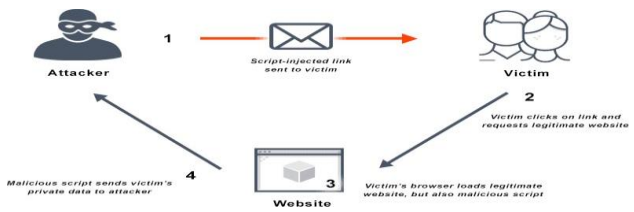
These scripts have the ability to perform unpleasant things like take your login credentials, create false content on websites, or even send you to phoney websites.
The script can play different pranks depending on where it's stored.

1. Reflected cross-site scripting: The script infiltrates content such as forum messages or search results. It automatically launches and has the ability to alter or steal data when the app delivers it back to your browser.

2. Stored XSS: The script here is concealed in user reviews or comments. This content is scripted to affect everyone who views it, which could result in content modifications or theft.

3. Based on DOM XSS: The script is hidden within the code of the website. The script tampers with the functionality of the page when you interact with specific areas of it, such as completing forms, which ultimately results in content modifications or data theft.

Developers must carefully handle user input, follow secure coding guidelines, and routinely scan for vulnerabilities in order to prevent becoming victims of cross-site scripting attacks (XSS).



C. Broken Access Control

When online applications mismanage user access privileges, they create a vulnerability known as Broken Access Control, which is exploited by attackers. This vulnerability poses significant security risks by enabling unauthorised individuals to access sensitive data or restricted functionalities.

Vulnerabilities related to broken access control can arise from a number of problems, including incorrect setup, insufficient authorization checks, and insufficient validation methods. Serious repercussions from these vulnerabilities could include account takeover, unauthorised data access, and system integrity compromise.

Attackers target Broken Access Control in a manner akin to how they inject malicious scripts via Cross-Site Scripting (XSS) to obtain unauthorised access. Attackers can escalate privileges and carry out unauthorised actions within the programme by tampering with access controls or evading authentication procedures.

Broken Access Control vulnerabilities come in several varieties:

1. Attackers can access unauthorised data or functionality by manipulating object references through Insecure Direct Object References (IDOR).
2. Lack of appropriate permission checks at the function level results in missing function level access control, which permits unauthorised access to particular functionality.
3. Attacks that use vulnerabilities to escalate privileges and obtain unauthorised access to higher-level functionalities are known as privilege escalation attacks.

To reduce Broken Access Control vulnerabilities, developers must carefully manage access restrictions, put safe authorization mechanisms in place, and carry out frequent security audits. If these vulnerabilities are not fixed, there may be major security lapses that jeopardise the availability, confidentiality, and integrity of sensitive data stored in the application.

III. Mitigating Web-Based Vulnerabilities

1. ***Understanding Web-Based Vulnerabilities***

Web-based vulnerabilities refer to weaknesses or flaws in web applications and servers that can be exploited by attackers to gain unauthorized access, steal sensitive data, or disrupt service. These vulnerabilities can arise from various sources, including poor coding practices, inadequate security policies, and the inherent complexity of web technologies.

2. Mitigation Strategies

The mitigation of web-based vulnerabilities involves several layers of security measures, from the coding phase to the deployment and maintenance phases. Below are detailed strategies to address these vulnerabilities:

2.1. Secure Coding Practices

Secure coding practices are essential to prevent vulnerabilities from being introduced in the development phase. This includes:

- **Input Validation:** Ensure that all input received from users is validated for type, length, format, and range. This can prevent injection flaws and other input-related vulnerabilities.
- **Output Encoding:** Properly encoding output data before sending it to the browser can mitigate the risk of XSS attacks.
- **Use of Prepared Statements and Parameterized Queries:** These practices should be used to prevent SQL injection attacks.
- **Regular Code Reviews:** Conducting regular code reviews and utilizing static and dynamic analysis tools can help detect and rectify security flaws before production.

2.2 Authentication and Session Management

Secure authentication and robust session management are vital for protecting user identity and data:

- **Multi-factor Authentication**: Implementing multi- factor authentication provides an additional security layer, making it harder for attackers to gain unauthorized access.

- **Session Management**: Secure handling of session tokens, including generation, renewal, and destruction, can prevent session hijacking.
- **Password Policies:** Enforcing strong password policies (complexity, expiration, history) enhances security.

2.3 Security Configurations

Proper configuration of web servers, databases, and software frameworks minimizes the surface for attacks:

- **Least Privilege Principle:** Systems should operate with the least set of privileges necessary to complete the job, reducing the potential damage from a breach.
- **Disable Unused Services:** Turning off services that are not needed minimizes potential entry points for attackers.
-**Secure Default Settings:** Always change default configurations to secure settings, including passwords, error messages, and file permissions.

2.4 Regular Updates and Patch Management

Keeping all systems up-to-date with the latest security patches is critical:

- **Patch Management Program:** Establish a rigorous program to regularly update all software components and dependencies.

- **Vulnerability Scanning and Penetration Testing:** Regularly scan the environment for vulnerabilities and conduct penetration tests to identify weak spots.

2.5. Data Protection

Ensuring data confidentiality and integrity involves several practices:

- **Encryption:** Use strong encryption protocols for data at rest and in transit.
-**Data Masking:** Mask data to protect sensitive information from unauthorized access, especially in development and testing environments.

2.6. Monitoring and Response

Continuous monitoring of web applications and infrastructure is essential to detect and respond to security incidents swiftly:

- **Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS):** Deploy these systems to monitor network traffic for suspicious activity.

- **Security Information and Event Management (SIEM):** Use SIEM systems to aggregate logs for advanced threat detection and real-time analysis.

Mitigating web-based vulnerabilities requires a multifaceted approach that includes secure coding practices, robust authentication mechanisms, vigilant security configurations, regular updates, vigilant monitoring, and responsive incident handling. By implementing these strategies, organizations can significantly reduce their exposure to web-based security threats and protect their users' data and privacy.

IV. CONCLUSION

In conclusion, our comprehensive study of web-based vulnerabilities underscores the critical need for a proactive and layered approach to cybersecurity. We have identified a variety of vulnerabilities that pose significant risks to web applications and outlined effective mitigation strategies to counteract these threats. Key to securing web applications is not only the implementation of advanced technical measures such as secure coding practices, rigorous authentication mechanisms, and continuous security assessments but also fostering a culture of security awareness across all levels of an organization.

The strategies discussed demonstrate that a holistic approach, integrating various security controls, is essential to develop a resilient web presence. As technology evolves and cyber threats become more sophisticated, it is imperative that organizations not only apply these current best practices but also remain vigilant and adaptive to new security challenges. This study highlights the importance of continuous improvement and innovation in cybersecurity measures to protect against emerging threats and ensure the integrity and confidentiality of user data in the digital age.

V. REFERENCES

1. Abd-El-Hafiz, S.K.; AbdElHaleem, S.H.; Radwan, A.G. Novel permutation measures for image encryption algorithms. *Opt. Lasers Eng.* **2016**, *85*, 72–83. [CrossRef]
2. Mahdi, M.S.; Azeez, R.A.; Hassan, N.F. A proposed lightweight image encryption using ChaCha with hyperchaotic maps. *Period. Eng. Nat. Sci.* **2020**, *8*, 2138–2145.
3. Liu, Y.; Jiang, Z.; Xu, X.; Zhang, F.; Xu, J. Optical image encryption algorithm based on hyper-chaos and public-key cryptography.
   Opt. Laser Technol. **2020**, *127*, 106171. [CrossRef]
4. Akkasaligar, P.T.; Biradar, S. Medical image encryption with integrity using DNA and chaotic map. In Proceedings of the International Conference on Recent Trends in Image Processing and Pattern Recognition, Solapur, India, 21–22 December 2018;
   pp. 143–153.
5. Belazi, A.; El-Latif, A.A.A.; Belghith, S. A novel image encryption scheme based on substitution-permutation network and chaos.
   Signal Process. **2016**, *128*, 155–170. [CrossRef]
6. Erkan, U.; Toktas, A.; Toktas, F.; Alenezi, F. 2D eπ-map for image encryption. *Inf. Sci.* **2022**, *589*, 770–789. [CrossRef]
7. Lai, Q.; Hu, G.; Erkan, U.; Toktas, A. A novel pixel-split image encryption scheme based on 2D Salomon map. *Expert Syst. Appl.*
   2003, *213*, 118845. [CrossRef]
8. Farah, M.A.B.; Farah, A.; Farah, T. An image encryption scheme based on a new hybrid chaotic map and optimized substitution box. *Nonlinear Dynam.* **2020**, *99*, 3041–3064. [CrossRef]
9. Ye, G.; Pan, C.; Huang, X.; Mei, Q. An efficient pixel-level chaotic image encryption algorithm. *Nonlinear Dyn.* **2018**, *94*, 745–756. [CrossRef]
10. Cai, Q. A secure image encryption algorithm based on composite chaos theory. *Treat. Du Signal* **2019**, *36*, 31–36. [CrossRef]
11. Wang, X.; Chen, X. An image encryption algorithm based on dynamic row scrambling and zigzag transformation. *Chaos Solitons Fractals* **2021**, *147*, 110962. [CrossRef]
12. Zhang, X.; Wang, L.; Cui, G.; Niu, Y. Entropy-based block scrambling image encryption using des structure and chaotic systems.
    Int. J. Opt. **2021**, *2019*, 3594534. [CrossRef]
13. Zhao, J.; Wang, S.; Zhang, L. Block Image Encryption Algorithm Based on Novel Chaos and DNA Encoding *Information* **2023**, *14*(3), 150. [CrossRef]
14. Jirjees, S.W. Alkalid, F.F.; Shareef, W.F. Image Encryption Using Dynamic Image as a Key Based on Multilayers of Chaotic Permutation. *Symmetry* **2023**, *15*(2), 409;[CrossRef]
15. Pareschi, F.; Rovatti, R.; Setti, G. On statistical tests for randomness included in the NIST SP800-22 test suite and based on the binomial distribution. *IEEE Trans. Inf. Foren. Secur.* **2012**, *7*, 491–505. [CrossRef]
16. D. H. Elkamchouchi, H. G. Mohamed, and K. H. Moussa, "A bijective image encryption system based on hybrid chaotic map diffusion and DNA confusion," Entropy, vol. 22, no. 2, 2020, doi: 10.3390/e22020180.
17. A. Girdhar and V. Kumar, "A RGB image encryption technique using Lorenz and Rossler chaotic system on DNA sequences," Multimed. Tools Appl., vol. 77, no. 20, pp. 27017–27039, 2018, doi: 10.1007/s11042-018-5902-z.
18. Xu, Ge et al. "Rational selection of RGB channels for disease classification based on IPPG technology." *Biomedical optics express* vol. 13,4 1820-1833. 3 Mar. 2022, doi:10.1364/BOE.451736.
19. Lamiaa Abdel-Hamid, Ahmed El-Rafei, Salwa El-Ramly, Georg Michelson, Joachim Hornegger, "Retinal image quality assessment based on image clarity and content," J. Biomed. Opt. 21(9),
    096007 (2016), doi: 10.1117/1.JBO.21.9.096007.