Advanced Research on Information Systems Security

ISSN: 2795-4609 | ISSN: 2795-4560
Print & Online

Cybersecurity Threats for a Web Development

João Pedro Cunha *

ISLA – Polytechnic Institute of Management and Technology, Vila Nova de Gaia, Portugal
Email: josopedro2015@hotmail.com

Abstract

With the increasing digitization of the world, web development has become an essential part of businesses and organizations worldwide. However, with the rapid development of technology, cyber threats and attacks have become a major concern for web developers. This article looks at some of the most common cybersecurity threats that web developers should be aware of, and the importance of taking preventative measures to secure web applications. By understanding these cyber threats and taking proactive steps to protect against them, web developers can ensure the security of their users' data and maintain the integrity of their web applications.

Keywords: web development, cybersecurity, vulnerabilities, owasp

* Corresponding author. Email address: josopedro2015@hotmail.com

1. Introduction

Technology is constantly evolving, and with it, cybercriminals are becoming more adept at developing new and advanced methods of attack. As a web developer, it is imperative to have a comprehensive understanding of current web threats in order to prevent or quickly respond to them [21], [26]. While some threats are commonplace, others can be new and complicated. To effectively protect websites and applications, it is important to know the latest threats and security measures. Therefore, it is advisable to stay informed about all the latest threats in order to stay alert and prepared for possible security breaches [29], [32].

It is important to recognize that no code is immune from threats. Hackers can bypass any code, security protocols, firewalls or other protections and gain unauthorized access to your application. Therefore, continuous monitoring, frequent threat scans, and adaptability are essential to stay ahead of potential security breaches. Staying vigilant and responding to new threats is necessary to ensure optimal protection of your application [29].

In this paper we will address the most important issues you need to know as a web developer [1], [2], [3], [4], [5].

1.1. Why is important Cybersecurity Web Development

Cybersecurity is essential not only for web development, but also for protecting digital assets, including sensitive information, from unauthorized access, theft or damage. It can protect companies and individuals from significant financial loss, reputational damage, and legal liability. Demonstrating proper data protection and security can avoid legal consequences and maintain customer trust.

Especially in web development, secure web applications have become increasingly important in light of the growing cybersecurity threats in today's digital landscape [1], [2], [3], [4], [5].

2. Top Security Threats

2.1. Sensitive Data Exposure

Sensitive data compromise is a critical security vulnerability that can occur when web applications do not adequately protect sensitive information. Sensitive data compromise is every organization's worst fear. Unfortunately, data breaches are common in all industries, and all types of businesses are potential targets. These breaches can lead to theft of personal information, financial fraud, and identity theft. [6]

2.1.1 How to prevent Sensitive Data Exposure

To prevent sensitive data exposure, web developers should use secure transmission protocols like HTTPS, encrypt sensitive data at rest using strong encryption algorithms, and properly manage encryption keys. Access to sensitive data should be limited through access control mechanisms and user authentication protocols.

Secure coding practices, such as input validation, sanitization, and encoding output, should be implemented to prevent common attack vectors such as injection attacks and cross-site scripting. Regular security testing, including vulnerability scanning, penetration testing, and security code reviews, should be conducted to identify and remediate vulnerabilities related to sensitive data exposure.

By following these best practices and conducting regular security testing, web developers can help prevent sensitive data exposure and ensure that their applications are secure and protected [6].

2.2. Broken Access Control

Broken Access Control refers to a security vulnerability that occurs when a system or application does not properly enforce the restrictions that users can access or change. Access control is a fundamental security principle that us responsible determining who can access what resources or data and what actions they can perform with those resources.

If access controls are broken, it can lead to unauthorized access, modification, or deletion of sensitive data or resources. Attackers can bypass access controls to gain administrative privileges, access sensitive data, or change critical system settings.

Access control vulnerabilities can take several forms, including improper authentication, insufficient privilege checks, privilege escalation, and direct object reference. Attackers can exploit these vulnerabilities to gain unauthorized access, modify or delete important data, and elevate their privileges [2], [7], [22], [23].

2.2.1 How to prevent Broken Access Control

To avoid the risk of inadequate access control, it is important to implement strong authentication and authorization mechanisms, ensure that users are granted only the minimum privileges necessary to perform their tasks, and validate all user input to prevent attacks [2], [7].

2.3. Cryptographic Failures

Cryptographic failure is a symptom rather than a cause. Cryptographic failures can result from a variety of factors, including weak encryption algorithms, insecure key management, improperly configured encryption, and cryptographic side-channel attacks, [24], [25].

Weak encryption algorithms can be easily compromised by attackers, while insecure key management can lead to theft or compromise of encryption keys.

Improperly configured encryption can allow attackers to intercept and decrypt communications, while cryptographic side-channel attacks exploit vulnerabilities in cryptographic implementations to obtain sensitive information [26], [28].

In some cases, cryptographic errors can also be the result of a deliberate attack, such as a cryptanalysis attack or a side-channel attack that exploits vulnerabilities in the cryptographic system to gain unauthorized access to sensitive data or resources. Therefore, while cryptographic flaws can be a symptom of deeper security problems, they can also be a direct and significant cause of security breaches and must be addressed accordingly [2], [8], [9].

2.3.1 How to Prevent Cryptographic Failures

To prevent the risk of cryptographic errors is important to use strong and up-to-date encryption algorithms, use

established cryptographic standards, manage encryption keys carefully, store sensitive data only if it's absolutely necessary and configure encryption protocols properly [2], [8], [9].

2.4. Injection

Injection is a vulnerability that allows an attacker to transmit malicious code or commands through another computer program or system.

Injection attacks can take many forms, such as SQL injection, command injection, and cross-site scripting (XSS) attacks, and can lead to a variety of security breaches, such as data loss or corruption, unauthorized access, and denial-of-service (DoS) attacks. Preventing injection attacks is an important part of ensuring the security and integrity of computer systems and applications [27], [30].

OWASP tests show that 94% of applications suffer some form of injection, with a maximum incidence rate of 19% [2], [10].

2.4.1 How to Prevent Injection

To prevent injection vulnerabilities, it is important to validate all user input and use parameterized queries in SQL statements, as well as apply safe coding practices such as input validation, error handling, and exception handling [2], [10].

2.5. Insecure design

Insecure design refers to a type of design or architecture that does not have adequate security measures or is vulnerable to security threats. It can refer to a variety of designs, including software, hardware, and infrastructure. An insecure design can make it easier for attackers to exploit vulnerabilities and gain unauthorized access to systems. Examples of insecure design include (but are not limited to) weak passwords, lack of encryption, lack of input validation and insecure communication protocols [2], [11].

2.5.1 How to Prevent Insecure design

To prevent an insecure design, it is important to follow security best practices and design principles from the beginning of the development process. It is possible to prevent these threats with methods such as using threat models, performing security testing, using secure coding practices, and establishing a secure development lifecycle [2], [11].

2.6. Identification and Authentication Failures

Identification and authentication failures refer to security incidents in which a system or application fails to correctly verify the identity of a user or device. These errors can lead to unauthorized access to sensitive data or systems and therefore pose a significant security risk. Identification and authentication failures can be caused by weak or easily guessed passwords, reuse of credentials, lack of multi-factor authentication, and susceptibility to

brute force or credential stuffing attacks [2], [12], [13].

2.6.1 How to Prevent Identification and Authentication Failures

To prevent identification and authentication failures, organizations should implement strong authentication mechanisms, such as multi-factor authentication and password policies that require users to choose strong passwords [2], [12], [13].

2.7. Software and Data Integrity Failures

Software and Data Integrity Failures refer to security incidents in which the integrity, accuracy, or availability of software or data is compromised. These errors can be caused by a variety of factors, including software bugs, malware infections, human error, and hardware failures. An example is when an application depends on libraries, plugins, or modules that come from untrusted sources, repositories, or content delivery networks.

Organizations must also be aware that software and data integrity failures can have significant legal and financial consequences. With increasing regulatory requirements related to data protection and privacy, companies that fail to prevent data breaches and other errors face stiff penalties and lawsuits [2], [14].

2.7.1 How to Prevent Software and Data Integrity Failures

To ensure the Software and Data Integrity, it's advisable to use digital signatures or similar mechanisms to verify that they come from trusted sources and have not been altered. In addition, it is important to ensure that libraries and dependencies use trusted repositories, with the option of hosting an internal repository of known quality for those with a higher risk profile.

A software supply chain security tool should also be used to verify that components do not contain known vulnerabilities.

A review process for code and configuration changes should be implemented to minimize the risk of malicious code or configurations being introduced into the software pipeline.

The CI/CD pipeline should have adequate separation, configuration, and access control to ensure code integrity during the build and deployment process.

Finally, it is important to never send unsigned or unencrypted serialized data to untrusted clients without integrity checking or digital signatures to prevent tampering or replay incidents. prevent identification and authentication failures, organizations should implement strong authentication mechanisms, such as multi-factor authentication and password policies that require users to choose strong passwords [2], [14].

2.8. Security Logging and Monitoring Failures

Security Logging and Monitoring Failures can have serious consequences for an organization's security posture.

Inadequate logging and monitoring can prevent the detection of security incidents, including security breaches and attacks, and affect an organization's ability to respond in a timely and effective manner.

Some of the vulnerabilities due to a lack of logging and monitoring can result in the failure to detect suspicious or alarming situations, the unavailability of locally logs in the event of a server failure, the loss of important information, or the difficulty of gaining useful insights [2], [15], [16].

2.8.1 How to Prevent Security Logging and Monitoring Failures

To avoid errors in security logging and monitoring, it is important to take effective measures. First, logging must be enabled for login and failed attempts, as well as for all other critical events. It is also important to have a powerful monitoring system that can detect suspicious or alarming events and alert security personnel. In addition, logs should be stored securely, preferably on a separate server or in the cloud, to prevent loss in the event of a server failure. The integrity of logging and monitoring systems should also be protected to prevent tampering or falsification by unauthorized individuals. Finally, regular log review and analysis can provide valuable insight and help identify potential security threats before they escalate [2], [15], [16].

2.9. Server-Side Request Forgery (SSRF)

Server-Side Request Forgery (SSRF) is a vulnerability that allows an attacker to make unauthorized requests on behalf of a vulnerable server. It occurs when a server-side application takes user input, such as a URL, and uses it to make requests to other resources.

Attackers can manipulate the user input to instruct the server to make requests to unauthorized resources, such as internal databases or sensitive systems. The consequences of an SSRF attack can be severe, such as data breaches, disruption of services, and unauthorized access to sensitive information.

In other cases, the attackers may connect to arbitrary external systems, allowing sensitive data such as credentials to be intercepted. The severity of SSRF is increasing due to the complexity of architectures and the widespread adoption of cloud services [2], [17].

2.9.1 How to Prevent Server-Side Request Forgery (SSRF)

To prevent SSRF attacks, server-side applications should carefully validate all user input and limit the resources that the server can access.

Administrators should ensure that their servers are securely configured and have limited access to internal resources and restricted network access. For example, network access control rules should be set up to block all

web traffic except important internal traffic, and firewall settings should be configured to "deny by default".

It is also recommended to isolate remote access functions on separate networks [2], [17].

2.10. Security Misconfiguration

Security misconfiguration is a vulnerability caused by incorrect security settings that can occur at any level of the technology stack, including network infrastructure, applications, and databases. It can leave systems and applications vulnerable to attack, with attackers exploiting various forms such as default passwords, unpatched systems, open ports, and unnecessary services or functions to gain unauthorized access, steal confidential information, or compromise system integrity [2], [18], [19].

2.10.1 How to Prevent. Security Misconfiguration

To avoid security misconfigurations, it is important to prioritize security and ensure that all security settings are properly configured and implemented. Some ways to avoid security misconfigurations include developing a repeatable patching plan, updating software, disabling default accounts, encrypting data, enforcing strict access controls, providing administrators with a repeatable process to avoid overlooking items, setting security settings in development frameworks to a safe value, and performing security scans and regular system audits. [18], [19]

2.11. Vulnerable and Outdated Components

Vulnerable and Outdated components refer to software components (e.g., libraries, third-party frameworks, etc.) that are used in an application but are no longer supported or have known vulnerabilities. These components can pose security risks to the application and its users, as attackers can exploit these vulnerabilities to compromise the application or steal sensitive data [2], [20].

2.11.1 How to Prevent Vulnerable and Outdated Components

Keeping software components up to date is critical to ensuring the security of an application, and developers should frequently check for updates to the components used in their applications, as new versions often fix known security vulnerabilities and other issues. If a component is no longer supported, developers should replace it with an alternative or remove it altogether. In addition to updating software components, developers should also consider using automated tools to scan their applications for security vulnerabilities and outdated components [2], [20].

3. Conclusion

In summary, web development faces significant cybersecurity threats as organizations increasingly rely on the Internet to operate and store sensitive data, leading to an increased risk of cyberattacks. Although web development and frameworks are becoming more secure, attackers are constantly finding new ways to exploit vulnerabilities. Therefore, deployment is not the end of the story, and software professionals must find and eliminate all potential vulnerabilities, which depends on their awareness of cyber threats and adherence to robust security practices. And the future will bring new changes for web development. While Web3 is becoming more popular, there are concerns about the security implications for Web development. The decentralized nature of Web3 poses new challenges for developers in terms of secure data storage and access control. To address these concerns, developers need to be familiar with the latest security best practices and ensure that their Web3 applications are thoroughly tested from the start and employ best practices.

References

[1] Natalie Paskoski " Top 10 Web Application Security Risks" [Online]: https://rhisac.org/application-security/top-ten-web-application-security-risks/, Nov. 3, 2022 [Fev. 10, 2023].

[2] Anna Dziuba "10 Common Web Application Security Vulnerabilities and How to Prevent Them in 2023" [Online]: https://relevant.software/blog/web-application-security-vulnerabilities/, Jan. 27, 2023 [Fev. 10, 2023].

[3] MDN contributors. " Website security" [Online]: https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Website_security, Feb. 25, 2023 [Fev. 10, 2023].

[4] Paweł Malita. "15 Critical Security Tips for Web Development in 2023" [Online]: https://www.netguru.com/blog/web-development-security, Jan. 28, 2023 [Fev. 10, 2023].

[5] S. K. Lala, A. Kumar and S. T., "Secure Web development using OWASP Guidelines," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2021, pp. 323-332, doi: 10.1109/ICICCS51141.2021.9432179.

[6] David Lukic "7 Cybersecurity Threats You Must Know as a Web Developer" [Online]: https://www.codemotion.com/magazine/backend/cybersecurity/cybersecurity-threats-web-developer/, Marc. 3, 2023 [Fev. 11, 2023].

[7] OWASP "A01:2021 – Broken Access Control" [Online]: https://owasp.org/Top10/A01_2021-Broken_Access_Control/, 2023 [Fev. 12, 2023].

[8] OWASP "A02:2021 – Cryptographic Failures" [Online]: https://owasp.org/Top10/A02_2021-Cryptographic_Failures/, 2023 [Fev. 13, 2023].

[9] Omkar Hiremath "Introduction to Cryptographic Failures" [Online]: https://www.softwaresecured.com/introduction-to-cryptographic-failures/, Jul 2022 [Fev. 13, 2023].

[10] OWASP "A03:2021 – Injection" [Online]: https://owasp.org/Top10/A03_2021-Injection/,2023 [Fev. 14, 2023].

[11] OWASP "A04:2021 – Insecure Design" [Online]: https://owasp.org/Top10/A04_2021-Insecure_Design/, 2023 [Fev. 15, 2023].

[12] OWASP "A07:2021 – Identification and Authentication Failures" [Online]: https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/, 2023 [Fev. 16, 2023].

[13] Cyolo Team "Identification And Authentication Failures And How To Prevent Them" [Online]: https://cyolo.io/blog/identification-and-authentication-failures-and-how-to-prevent-them/, Marc. 11 2022 [Fev. 17, 2023].

[14] OWASP "A08:2021 – Software and Data Integrity Failures" [Online]: https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures/, 2023 [Fev. 18, 2023].

[15] Educative Team "What are security logging and monitoring failures?" [Online]: https://www.educative.io/answers/what-are-security-logging-and-monitoring-failures, Jan. 18 2023 [Fev. 19, 2023].

[16] OWASP "A09:2021 – Security Logging and Monitoring Failures [Online]: https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/, 2023 [Fev. 20, 2023].

[17] OWASP "A10:2021 – Server-Side Request Forgery (SSRF) [Online]: https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/, 2023 [Fev. 21, 2023].

[18] OWASP " A05:2021 – Security Misconfiguration [Online]: https://owasp.org/Top10/A05_2021-Security_Misconfiguration/, 2023 [Fev. 22, 2023].

[19] Tyra Appleby "A guide to preventing common security misconfigurations" [Online]: https://resources.infosecinstitute.com/topic/guide-preventing-common-security-misconfigurations/, Mar. 21 2018 [Fev. 23, 2023].

[20] OWASP " A06:2021 – Vulnerable and Outdated Components [Online]: https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/, 2023 [Fev. 24, 2023].

[21] Mateus-Coelho, Nuno Ricardo, et al. "POSMASWEB: Paranoid Operating System Methodology for Anonymous and Secure Web Browsing." Handbook of Research on Cyber Crime and Information Privacy, edited by Maria Manuela Cruz-Cunha and Nuno Ricardo Mateus-Coelho, IGI Global, 2021, pp. 466-497. https://doi.org/10.4018/978-1-7998-5728-0.ch023

[22] P. F. Wilbur, "HACKERNOON. A hacker intercepted your WiFi traffic, stole your contacts, passwords, & financial data.," 2 february 2019. [Online]. Available: https://hackernoon.com/a-hacker-intercepted-your-wifi-traffic-stole-your-contacts-passwords-financial-data-heres-how-4fc0df9ff152. [Accessed 2022 december 3].

[23] D. Schepers, A. Ranganathan and M. Vanhoef, "Let Numbers Tell the Tale: Measuring Security Trends in Wi-Fi Networks and Best Practices," in WiSec '21, Abu Dhabi, United Arab Emirates, 2021.

[24] Z. A. Najar and R. N. Mir, "Wi-Fi: WPA2 Security Vulnerability and Solutions," Wireless Engineering and Technology, vol. 22, pp. 15-22, 18 April 2021.

[25] Cisco, "What Is Wi-Fi Security?," [Online]. Available: https://www.cisco.com/c/en/us/products/wireless/what-is-wi-fi-security.html#~q-a. [Accessed 4 december 2022].

[26] Nuno Mateus-Coelho, A New Methodology for the Development of Secure and Paranoid Operating Systems, Procedia Computer Science, Volume 181, 2021, Pages 1207-1215, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2021.01.318

[27] A. Sharma, T. Bhatia, A. Katyar and S. E. U, "Wireless Security – An Introduction to Wireless Security Protocols and their Security Flaws," Annals of R.S.C.B., vol. 25, no. 6, pp. 11805-11812, 2021.

[28] J. L. MacMichael, "Auditing Wi-Fi Protected Access (WPA) Pre-Shared Key Mode," Linux Journal, 28 july 2005.

[29] N. Mateus-Coelho and M. Cruz-Cunha, "Serverless Service Architectures and Security Minimals," 2022 10th International Symposium on Digital Forensics and Security (ISDFS), Istanbul, Turkey, 2022, pp. 1-6, doi: 10.1109/ISDFS55398.2022.9800779.

[30] D. J. Fehér and B. Sándor, "Effects of the WPA2 KRACK Attack in Real Environment," in IEEE 16th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, 2018.

[31] E. Lamers and R. Dijksman, "Securing home Wi-Fi with WPA3 personal," in 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC), 2021.

[32] F. Alves, N. Mateus-Coelho and M. Cruz-Cunha, "ChevroCrypto – Security & Cryptography Broker," 2022 10th International Symposium on Digital Forensics and Security (ISDFS), Istanbul, Turkey, 2022, pp. 1-5, doi: 10.1109/ISDFS55398.2022.9800797.