IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Identification of Web Vulnerabilities through IISRA Framework

Keerti Dixit
Institute of Computer Science
Vikram University, Ujjain

Dr. Umesh Kumar Singh
Institute of Computer Science
Vikram University, Ujjain

Dr. Bhupendra Kumar Pandya
Institute of Computer Science
Vikram University, Ujjain

Abstract: Attackers frequently target web applications for a number of reasons, such as obtaining information from either the website's data storage or serving harmful files in place of legitimate ones. Web applications have a wide variety of known vulnerabilities which may be utilized to attack a website; it is crucial to find these prior to them being exposed. In this research paper, we have developed an Integrated Information Security Risk Assessment (IISRA) Framework for identification of Web Vulnerabilities. We have implemented this IISRA Framework on a website of an organization's real computing environment.

Keywords: Web Vulnerabilities, Risk, Threat, Impact.

1. Introduction

The widespread use of the Internet and the quick advancement of web application technology have made it possible for a growing number of web applications to be developed on the Internet as a foundational platform [1-6]. People's lifestyles have significantly changed as a result of popular online shopping complexes, internet banking, and other web apps [7–10]. Despite going anywhere they can conveniently go shopping or take care of financial issues. Unfortunately, these emerging innovations not only make work and even learning more convenient, but they also pose significant threats which we've not faced before. The bar for web application attack technique is falling as a result of the development of network technology. Attack objects are gradually moved from the computer network to the web - based application by hackers [11–14]. 75% of information security threats, as reported by Gartner's survey, take place on Web apps rather than at the network level. Two-thirds of online apps are determined to be extremely vulnerable to attacks at the same time. It is unfortunate that many businesses concentrate their efforts and financial resources on server & network security and ignore the security issues with web apps, providing a golden opportunity for attackers [15–17].

Web applications are vulnerable primarily because people can upload data at any time and without sufficient verification on the server-side. From the standpoint of the software itself, the primary cause is that the period of creating web apps is getting increasingly shorter and the expertise of programmers is unequal, which results in the

incomplete examination of vulnerabilities in the software development process. To maintain the security of the web service and avoid an attack, a thorough penetration test needs to be carried out on it prior to the hacker initiates their attack [16, 17]. By using web-based application penetration testing technology, we can recognize the vulnerabilities of web - based applications prior to being attacked, identify the level of vulnerability, effectively recognize web - based applications, modify malicious Web pages, transmit them to the server, and assess the presence of security vulnerabilities in web - based applications by analyzing the response given back by the server.

2. IISRA Framework for Web Vulnerabilities Identification

We have developed an Integrated Information Security Risk Assessment (IISRA) Framework for identification of web vulnerabilities. IISRA Framework helps in identifying and assessing potential security vulnerabilities in web applications.

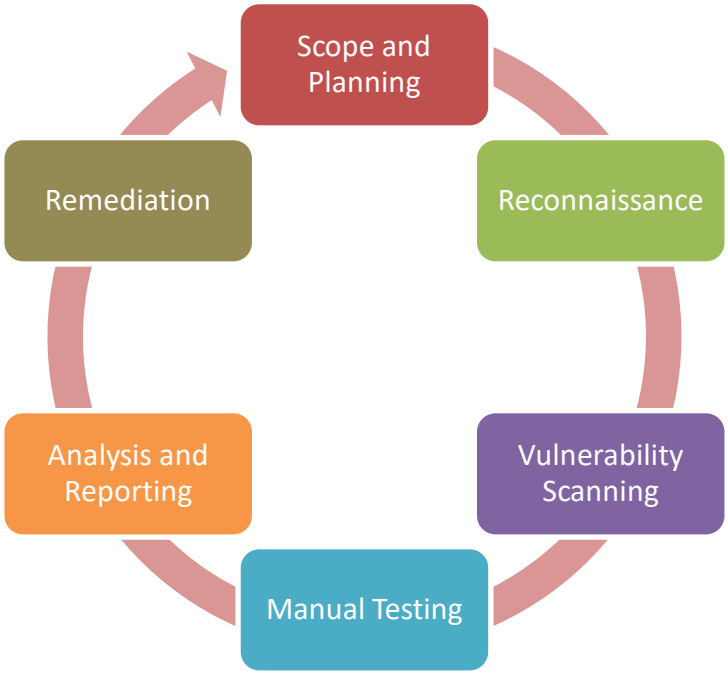

Figure 1: IISRA Framework for Web Vulnerabilities Identification

The following are the general steps involved in IISRA Framework:

Scope and Planning: The first step is to define the scope of the web application to be scanned and plan the scanning process. This involves identifying the target web application, defining the objectives of the scan, and selecting the appropriate scanning tools.

Reconnaissance: The next step is to perform reconnaissance on the target web application to gather information about the application and its underlying technology stack. This involves using techniques such as port scanning, banner grabbing, and web application fingerprinting.

Vulnerability Scanning: Once the reconnaissance is complete, the vulnerability scanning process can begin. This involves using automated scanning tools to identify potential vulnerabilities in the target web application. The

scanning tools use various techniques such as crawling the web application, sending crafted HTTP requests, and analyzing the responses to identify potential vulnerabilities.

Manual Testing: Automated scanning tools can only identify a limited set of vulnerabilities. To identify more complex vulnerabilities, manual testing is required. This involves conducting in-depth testing of the web application using various techniques such as penetration testing, input validation testing, and business logic testing.

Analysis and Reporting: After the scanning and testing process is complete, the results are analyzed and reported to the stakeholders. The report should contain a detailed description of the vulnerabilities identified, their severity, and recommendations for mitigating the vulnerabilities.

Remediation: The final step is to remediate the vulnerabilities identified in the scan. This involves fixing the vulnerabilities and implementing measures to prevent similar vulnerabilities from occurring in the future.

3. Results and Mitigation Plan

We have implemented IISRA Framework in the real scenario of an organization to assess the web vulnerabilities of that organization. Below table illustrates distribution of observations of application security assessment based onthe risk categorization i.e., Critical, High, Medium, and Low.

Table 1: Results of Web Vulnerability Risk Assessment

Domain	Critical	High	Medium	Low	Total
Web Application	0	1	1	7	9

The chart given below represents the vulnerabilities found during web application security assessment:

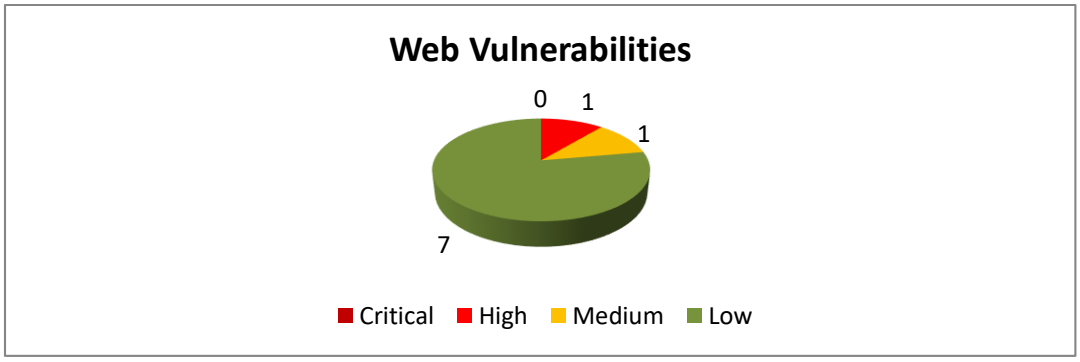

Figure 2: Web Vulnerability Assessment Test Observations

An overview of the observations made during the web application vulnerability assessment process is provided in this section. The comprehensive list of observations, together with each observation's specific risk classification and associated actions to mitigate the risks, are provided in the detailed observations section.

Table 2: Web Application Risk Assessment and Mitigation Plan through IISRA Framework

Vulnerabilities	Impact	OWASP Mapping	Risk	Recommendations
Weak Password policy And No-Account Lockout	In order to give an assertion of identity for a system user, authentication systems frequently depend on a secret that is memorised (also known as a password). It is crucial that this password be sufficiently complicated and difficult for an enemy to guess. Moreover, no account lockout capability was implemented by the application when the wrong password was input repeatedly.	A5	High	It is recommended to mitigate the risk of easily guessed passwords facilitating unauthorized access there are two solutions: introduce additional authentication controls (i.e., two-factor authentication) or introduce a strong password policy. The simplest of these is the introduction of a strong password policy that ensures password length, complexity, reuse, and aging; although ideally both should be implemented. It is also recommended to Implement Captcha and lockout after a defined number of incorrect passwords attempts to avoid this scenario.
SSL/TLS Diffie-Hellman Modulus <= 1024 Bits (Logjam)	TLS 1.0 and 1.1 are vulnerable to downgrade attacks since they rely on the SHA-1 hash to ensure the message integrity. Even handshake authentication is based on SHA-1, making it simpler for an attacker to pose as a server for MITM attacks. The newer protocols allow users to choose more secure hashing methods, whereas TLS 1.1 or earlier does not.	A2	Medium	It is recommended to reconfigure the service to use a unique Diffie-Hellman moduli of 2048 bits or greater.
Unhandled Error Message – Stack Trace and Platforms Data leaks	Based on the output gathered, displaying comprehensive error information can be used to launch targeted attacks on the server, giving an attacker access to data about your system.	A6	Low	It is recommended to have a custom error page which doesn't leak stack trace data.
Clickjacking	Using seemingly innocent objects, such as web pages, to fool a user into clicking	A5	Low	It is recommended to configure your web server to include an X-

	on something different from what they think they are doing might divulge sensitive information or let others to take control of their computer. This practise is known as clickjacking.			Frame-Options header and a CSP header.
Vulnerable JavaScript Dependency	Using JavaScript libraries from third parties can bring a number of DOM-based vulnerabilities, including those like DOM-XSS that can be leveraged to take over user accounts.	A6	Low	It is recommended to develop a patch-management strategy to ensure that security updates are promptly applied to all third-party libraries in your application. Also, consider reducing your attack surface by removing any libraries that are no longer in use.
Concurrent Login	Concurrent logins make it more difficult for a user to determine whether their account has been compromised because both unauthorised and authorised use may take place at the same time. Also, allowing a user to log in more than once may result in concurrency errors. These are the problems that occur when various requests from various sessions update the same data (nearly) simultaneously. Depending on the type of data being edited, this may result in inconsistencies or exceptions; at the very least, it may burn up extra resources, confuse the user, and provide inconsistent log entries.	A7	Low	It is recommended that user accounts within a web application should only be permitted to use one session at a time. If the user authenticates again then any previously valid sessions should be immediately terminated, with an appropriate message displayed within both sessions.
Version Disclosure via Banner Grabbing	This knowledge may then be put to use to create unique, server-specific payloads for other assaults.	A5	Low	It is recommended to configure the server in such a way that it leaks the least amount of information about the server.
Password Field with	An attacker who seizes control of the user's	A5	Low	It is recommended to prevent browsers from

Autocomplete Enabled	computer may be able to steal the saved credentials. Furthermore, a user's browser-stored credentials can be accessible to an attacker who discovers a different application vulnerability, such as cross-site scripting.			storing credentials entered HTML forms, include the attribute autocomplete="off" within the FORM tag (to protect all form fields) or within the relevant INPUT tags (to protect specific individual fields).
Improper Cookie Attribute	Failure to use the 'Lax' or 'Strict' values could raise the risk of vulnerability to CSRF attacks if the website does not enforce extra defence against CSRF attacks. A cookie should always be sent via an encrypted channel if it includes sensitive data or is a session token; otherwise, it can be retrieved through unencrypted connections.	A5	Low	It is recommended to set the Same Site attribute of a sensitive cookie to 'Lax' or 'Strict'. This instructs the browser to apply this cookie only to same-domain requests, which provides a good Defense in Depth against CSRF attacks. Also ensure that the secure flag is set for cookies containing such sensitive information.

4. Conclusion

In this research paper, we have assessed web vulnerabilities of an organization through IISRA framework. Web application of the organization has been tested against most of the application security related issues. During the application security testing, we have found that the website of the organization is vulnerable to nine security issues as on date tested. We observed one high, one medium and seven low vulnerabilities which needed attention. Finally we have suggested mitigation plan for the identified web vulnerabilities.

5. References

[1] Y. Dong, “Application of artificial intelligence software based on semantic web technology in English learning and teaching,” Journal of Internet Technology, vol. 23, no. 1, pp. 143–152, 2022.

[2] L. Luo, “Web application software engineering technology and process,” in Proceedings of the Ieee Asia-Pacific Conference On Image Processing, Electronics And Computers, Ipec 2021, pp. 935–937, Dalian, China, Apirl 2021.

[3] L. Li, B. Lei, and C. Mao, “Digital twin in smart manufacturing,” Journal of Industrial Information Integration, vol. 26, no. 9, Article ID 100289, 2022.

[4] H. Lin, “Application of web 2.0 technology to cooperative learning environment system design of football teaching,” Wireless Communications and Mobile Computing, vol. 2022, Article ID 5132618, 9 pages, 2022.

[5] R. Chen, “The design and application of college english-aided teaching system based on web,” Mobile Information Systems, vol. 2022, Article ID 3200695, 10 pages, 2022.

[6] L. Li, T. Qu, Y. Liu et al., “Sustainability assessment of intelligent manufacturing supported by digital twin,” IEEE Access, vol. 8, pp. 174988–175008, 2020.

[7] S. Qi, S. Li, and J. Zhang, “Designing a teaching assistant system for physical education using web technology,” Mobile Information Systems, vol. 2021, Article ID 2301411, 11 pages, 2021.

[8] L. Li, C. Mao, H. Sun, Y. Yuan, and B. Lei, “Digital twin driven green performance evaluation methodology of intelligent manufacturing: hybrid model based on fuzzy rough-sets AHP, multistage weight synthesis, and PROMETHEE II,” Complexity, vol. 2020, no. 6, 24 pages, Article ID 3853925, 2020.

[9] K. Zhang, “Web news data extraction technology based on text keywords,” Complexity, vol. 2021, Article ID 5529447, 11 pages, 2021.

[10] L. Li and C. Mao, “Big data supported PSS evaluation decision in service-oriented manufacturing,” IEEE Access, vol. 8, no. 99, pp. 154663–154670, 2020.

[11] J. Li, Y. Fu, J. Xu, C. Ren, X. Xiang, and J. Guo, “Web application attack detection based on attention and gated convolution networks,” IEEE Access, vol. 8, pp. 20717–20724, 2020.

[12] M. Babiker, E. Karaarslan, and Y. Hoscan, “Web application attack detection and forensics: a survey,” in Proceedings of the 6th International Symposium On Digital Forensic And Security, pp. 1–6, Antalya, Turkey, March 2018.

[13] S. Ninawe and R. Wajgi, “Detection of DOM-based XSS attack on web application,” Lecture Notes on Data Engineering and Communications Technologies, vol. 33, pp. 633–641, 2020.

[14] K. B. Jalbani, M. Yousaf, M. S. Sarfraz, J. Oskouei, A. Hussain, and Z. Memon, “Poor coding leads to dos attack and security issues in web applications for sensors,” Security and Communication Networks, vol. 2021, Article ID 5523806, 11 pages, 2021.

[15] X. Yu, W. Yu, S. Li, X. Yang, Y. Chen, and H. Lu, “WEB DDoS attack detection method based on semisupervised learning,” Security and Communication Networks, vol. 2021, Article ID 9534016, 10 pages, 2021.

[16] A. K. Dalai and S. K. Jena, “Neutralizing SQL injection attack using server side code modification in web applications,” Security and Communication Networks, vol. 2017, Article ID 3825373, 12 pages, 2017.

[17] R. R. Echeverria, J. C. Preciado, A. Rubio-Largo, ´ J. M. Conejero, and A. E. Prieto, “A pattern-based develop- ´ ment approach for interaction flow modeling language,” Scientific Programming, vol. 2019, Article ID 7904353, 15 pages, 2019.