Volume 11, Issue 3, March – 2026 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26mar854

Secure Scan: Threat Intelligence Platform

Aryan P. Kalbandhe[1]; Isha K. Pakhode[2]; Saurabh V. Bhatkar[3]; Aditya D. Barate[4]

Publication Date: 2026/03/23

Abstract: With the increasing use of web applications and online systems, cybersecurity threats such as malware and vulnerable code have become a major challenge for developers and organizations. Many security issues arise due to insecure coding practices or the use of unverified files and links. To address this problem, this project presents Secure Scan, a web-based security analysis tool designed to detect malicious files and identify common code vulnerabilities.

Secure Scan integrates the VirusTotal API to scan files and URLs using multiple antivirus engines, helping users quickly identify potentially harmful content. In addition, the system includes a code scanning module that analyzes source code for common security vulnerabilities based on the OWASP Top 10, such as SQL Injection, Cross-Site Scripting (XSS), and command injection. The backend of the system is developed using Node.js, which manages the scanning process, vulnerability detection logic, and API communication.

The goal of Secure Scan is to provide developers with a simple and accessible platform to perform basic security checks during the development process. By combining malware detection and code vulnerability analysis into a single tool, Secure Scan helps promote secure coding practices and improves overall software security.

How to Cite: Aryan P. Kalbandhe; Isha K. Pakhode; Saurabh V. Bhatkar; Aditya D. Barate (2026) Secure Scan: Threat Intelligence Platform. *International Journal of Innovative Science and Research Technology*, 11(3), 1873-1879. https://doi.org/10.38124/ijisrt/26mar854

I. INTRODUCTION

The rapid growth of web technologies and online services has significantly increased the dependency of organizations on web applications. From small businesses to large enterprises, web-based platforms are now used for communication, data management, financial transactions, and service delivery. While this digital transformation has improved efficiency and accessibility, it has also introduced serious cybersecurity challenges. Web applications often contain vulnerabilities that attackers can exploit to gain unauthorized access, steal sensitive information, or disrupt services. As a result, ensuring the security of web applications has become one of the most important concerns in modern software development. One of the widely recognized standards for identifying common web application vulnerabilities is the Open Web Application Security Project (OWASP) Top 10. This framework highlights the most critical security risks that developers and organizations should be aware of when building web applications. These risks include threats such as SQL injection, cross-site scripting (XSS), broken authentication, insecure design, security misconfigurations, and exposure of sensitive data. The OWASP Top 10 serves as a guideline for developers and security professionals to understand the major weaknesses that commonly occur in web systems. However, identifying these vulnerabilities manually can be time-consuming and difficult, especially for developers who may not have extensive knowledge of cybersecurity practices. In addition to insecure code, another major threat faced by modern systems is the presence of malicious files and unsafe URLs. Attackers often distribute malware through infected files, malicious downloads, or harmful links embedded in web pages. If such files are executed within a system, they can compromise the entire environment by installing malicious software, stealing data, or creating unauthorized backdoors. Therefore, detecting potentially harmful files and links before they are executed is an essential step in improving system security.

To address these challenges, this paper introduces Secure Scan, a web-based security analysis platform designed to assist developers and users in identifying potential security threats. The main objective of Secure Scan is to combine multiple security analysis techniques into a single system that can detect both **and** malware threats code vulnerabilities. By integrating these functionalities, the platform provides a simple yet effective approach for improving security awareness and detecting risks during the development process.

The Secure Scan platform includes two major components: malware scanning and source code vulnerability detection. The malware scanning feature uses the VirusTotal API, which allows files and URLs to be analyzed using multiple antivirus engines and security databases. This integration enables the system to detect malicious or suspicious files by comparing them against a large collection of known threat signatures. When a user uploads a file or submits a URL, the platform sends the data to the VirusTotal service and retrieves a detailed analysis report that indicates whether the content is safe or potentially harmful. Along with malware detection, Secure Scan also includes a code scanning module designed to identify common security vulnerabilities within source code. The module analyzes uploaded code files and searches for patterns that may indicate security weaknesses related to the OWASP Top 10 categories. For example, the system can detect indicators of SQL injection, cross-site scripting, command injection, and the use of weak cryptographic algorithms. By automatically highlighting these issues, the platform helps developers recognize potential security flaws and encourages them to follow safer coding practices.

The system is implemented using Node.js as the backend technology. Node.js provides an event-driven and non-blocking architecture, which allows the platform to handle multiple scanning requests efficiently. This makes the system suitable for real-time analysis and responsive user interactions. The platform also includes a user-friendly interface that allows users to easily upload files, submit URLs, and analyze source code while viewing detailed scan results in a clear and understandable format.

The purpose of this research is to design and evaluate a lightweight yet effective security analysis platform that assists developers in identifying potential threats at an early stage. By combining malware detection with code vulnerability analysis, Secure Scan aims to simplify the security testing process and promote better cybersecurity practices among developers and students. Overall, this study demonstrates how integrating automated scanning tools with established security frameworks such as the OWASP Top 10 can help improve the security of web applications. The Secure Scan platform provides a practical example of how modern technologies and external threat intelligence services can be combined to create accessible security solutions for real-world applications.

II. LIMITATIONS IN PREVIOUS WORKS

Although various cybersecurity tools and research studies have attempted to improve the detection of web application vulnerabilities, several limitations still exist in existing systems. Many available solutions focus on only one aspect of security, such as vulnerability scanning or malware detection, rather than providing a unified platform that addresses multiple security threats simultaneously. These limitations highlight the need for a more practical and integrated solution like **Secure Scan**, which combines different security analysis techniques into a single system.

➢ *Lack of Integrated Malware and Code Vulnerability Detection*

Many existing security tools focus on either malware detection or source code vulnerability analysis, but rarely combine both functionalities within the same platform. For example, malware detection services such as VirusTotal specialize in analyzing files and URLs for malicious content using multiple antivirus engines. However, these platforms do not analyze the security of the source code used in web applications. Similarly, vulnerability scanners such as OWASP ZAP primarily focus on identifying weaknesses in running web applications rather than detecting malicious files. This separation of tools makes it difficult for developers to perform complete security analysis in a single environment.

➢ *Limited Accessibility for Developers and Students*

Another limitation observed in many existing solutions is that they are often designed for professional penetration testers or security experts. Tools such as Burp Suite provide advanced security testing capabilities but may require significant technical knowledge to operate effectively. As a result, beginner developers or students may find it difficult to use these tools for learning or basic security testing. This gap highlights the need for simplified platforms that can help developers easily understand common vulnerabilities and security risks during the development process.

➢ *Lack of Automated Code Vulnerability Identification*

Many development environments still rely on manual code review to detect security vulnerabilities. This approach can be time-consuming and may fail to detect subtle security flaws present in the code. While some tools provide automated vulnerability scanning, they are often complex or require extensive configuration. Furthermore, these tools may not specifically focus on common security risks defined in the OWASP Top 10, which are widely recognized as the most critical web application vulnerabilities. Without automated detection mechanisms, developers may unintentionally deploy applications containing insecure coding practices.

➢ *Performance Limitations in Traditional Security Tools*

Some existing security solutions are built on heavy frameworks or require high computational resources to perform vulnerability analysis. Such systems may struggle to handle multiple requests or real-time scanning efficiently. Modern web applications require security tools that can process large amounts of data quickly and provide immediate feedback. Technologies such as Node.js offer an event-driven architecture that allows systems to handle multiple operations simultaneously, but many previous implementations have not fully utilized these capabilities to build lightweight and responsive security platforms.

➢ *Lack of User-Friendly Security Analysis Platforms*

Many security tools provide detailed reports and technical outputs that can be difficult for users to interpret. These reports often contain large amounts of raw data without clearly explaining the severity or impact of detected vulnerabilities. As a result, developers may find it challenging to identify which issues require immediate attention. A user-friendly interface with clear vulnerability descriptions and simple scan results can significantly improve the usability of security tools.

➢ *Need for a Unified Security Analysis Platform*

Due to the limitations mentioned above, developers often need to use multiple security tools to analyze different types of threats. This fragmented approach increases the complexity of the security testing process and reduces efficiency. Therefore, there is a need for a unified platform that integrates malware detection, vulnerability scanning, and easy-to-understand security reports within a single system. The **Secure Scan** project aims to address these limitations by providing a lightweight web-based platform that allows users to scan files, URLs, and source code while detecting common vulnerabilities and malicious threats in an accessible and efficient manner.

III. PROPOSED SYSTEM – SECURE SCAN

To overcome the limitations observed in existing security tools, this research proposes Secure Scan, a web-based security analysis platform designed to help developers and users identify potential cybersecurity threats in an efficient and accessible manner. The system focuses on detecting both malicious files and insecure source code by combining external threat intelligence services with automated vulnerability detection techniques. By integrating these functionalities within a single platform, Secure Scan aims to simplify the security analysis process and improve awareness of common cybersecurity risks.

The architecture of the proposed system is designed to support file scanning, URL analysis, and source code vulnerability detection through an interactive web interface. The platform allows users to upload files, submit URLs, or provide source code for analysis. The system is developed using Node.js, which provides an event-driven and non-blocking architecture. This design allows the platform to handle multiple scanning requests efficiently while maintaining good performance. Node.js also simplifies the integration of external APIs and backend processing logic, making it suitable for building scalable web security applications.

One of the core components of Secure Scan is the malware detection module, which integrates with VirusTotal. This service aggregates results from multiple antivirus engines and threat intelligence databases to analyze suspicious files and URLs. When a user uploads a file or submits a link, the system sends the data to the VirusTotal API. The API then performs security analysis and returns a report indicating whether the file or URL is considered safe, suspicious, or malicious. The results are processed by the backend and displayed to the user in an easy-to-understand format. In addition to malware detection, Secure Scan includes a source code vulnerability scanning module. This module analyzes uploaded code files and searches for patterns that may indicate insecure coding practices. The detection process is based on predefined rules that correspond to common vulnerabilities listed in the OWASP Top 10, which is a widely recognized guideline for identifying major web application security risks. Examples of vulnerabilities detected by the system include SQL injection, cross-site scripting (XSS), command injection, and insecure cryptographic implementations. By automatically identifying these issues, the platform assists developers in improving the security of their applications during the development phase.

Another important aspect of the proposed system is its user-friendly interface. The platform is designed to make security analysis accessible not only to experienced security professionals but also to developers and students who may have limited cybersecurity knowledge. The interface allows users to easily upload files, paste URLs, or submit source code and quickly receive scan results. The results are presented with clear descriptions of detected vulnerabilities and potential risks, helping users understand the security implications of their code or files. The workflow of Secure Scan can be summarized in several stages. First, the user provides input to the system in the form of a file, URL, or code snippet. Next, the backend processes the request and sends the input to the appropriate scanning module. After the analysis is completed, the results are processed and displayed to the user through the web interface. Below is the workflow of secure scan:

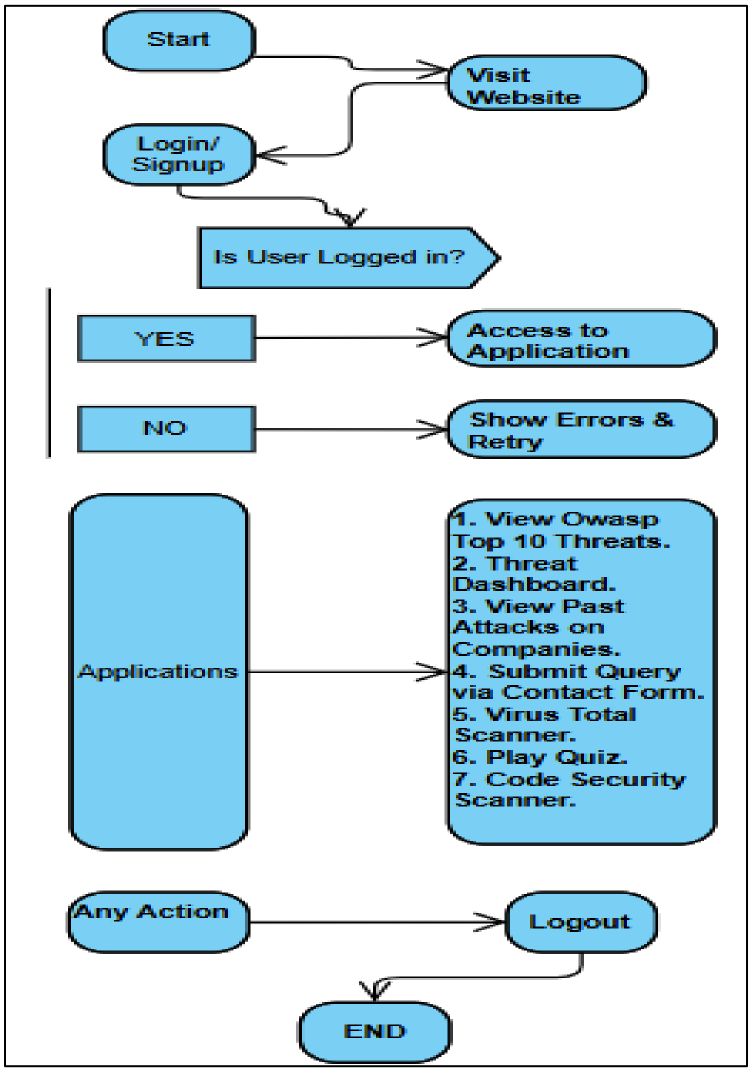


Fig 1: Work Flow

IV. FUNCTIONALITY SECTION

Here we used various domains and the Secure Platform is based on the OWASP Top 10 vulnerabilities, these vulnerabilities are as follows:

- ➢ *Broken Access Controls*
  Enforces strict access control policies to ensure users can only access resources and perform actions aligned with their permission.

- ➢ *Cryptographic Failure*
  Protects sensitive information through encryption, secure transmission protocols, and minimizing data storage to prevent unauthorized access.

- ➢ *Injection Flaws*
  Prevents malicious code execution by sanitizing user inputs and using parameterized queries to block unauthorized database commands.

- ➢ *Insecure Design*
  Validates and sanitizes serialized data to prevent remote code execution, privilege escalation, or tampering during object reconstruction.

- ➢ *Security Misconfiguration*
  Implements secure default configurations, regular updates, and vulnerability scanning to eliminate unnecessary exposure of system components.

- ➢ *Vulnerable and Outdated Component*
  Regularly updates and patches third-party libraries and frameworks to mitigate risks from outdated or vulnerable dependencies.

- ➢ *Identification and Authentication Failures*
  Ensures secure authentication mechanisms by enforcing strong password policies, multi-factor authentication, and session management.

➢ *Software and Data Integrity Failures*

It ensures the prevention of unauthorized code or data modifications by enforcing secure software supply chain practices, validating updates through digital signatures, and protecting sensitive data during transit and storage to maintain system reliability and trustworthiness.

➢ *Security Logging and Monitoring Failures*

It ensures comprehensive threat detection and response by implementing robust logging mechanisms, real-time monitoring, and actionable alert systems to identify, track, and mitigate suspicious.

➢ *Server-Side Request Forgery*

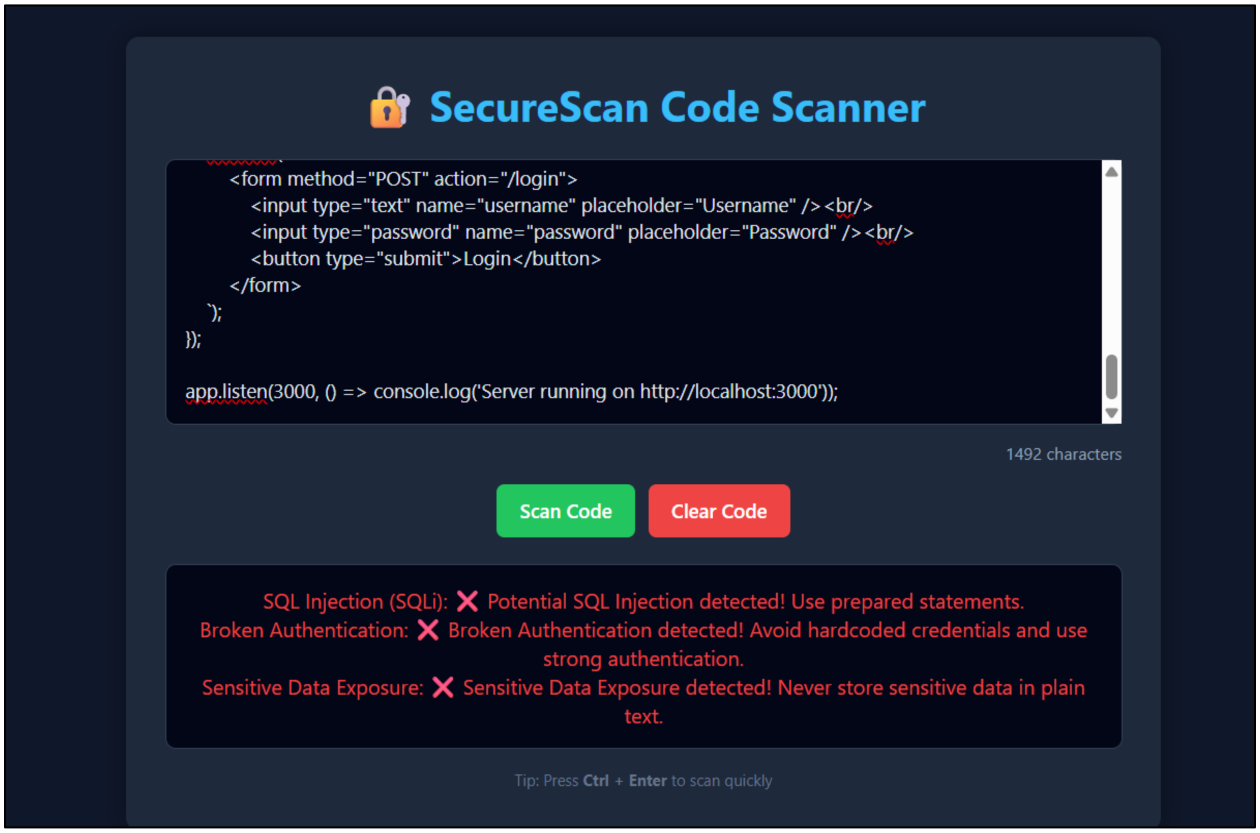
It prevents attackers from exploiting server requests to access internal systems or sensitive data by validating and restricting external resource calls, enforcing strict network controls, and sanitizing user-supplied input.

V. RESULT

The overall results indicate that Secure Scan provides a lightweight yet effective solution for basic cybersecurity analysis. By combining malware detection with automated vulnerability scanning, the platform helps bridge the gap between theoretical security knowledge and practical implementation. Furthermore, the modular design of the system allows future improvements such as integrating additional threat intelligence sources, expanding vulnerability detection techniques, and enhancing reporting features.

The development of this Secure Scan platform demonstrates a practical and comprehensive approach to cybersecurity education, effectively bridging theoretical knowledge with real-world application. By integrating OWASP Top 10 vulnerabilities into an interactive learning environment, the platform empowers users to understand, identify, and mitigate critical security threats through hands-on tools such as a virus scanner, code analyzer, and real-time dashboard tracking active and historical cyberattacks on organizations like Airtel and WazirX. The inclusion of Firebase for secure authentication ensures scalable user management, while the gamified quiz system enhances engagement, promoting knowledge retention and proactive threat-handling behavior. The platform’s clean interface and structured dashboard simplify complex threat data, making it accessible to both beginners and professionals.

In summary, the Secure Scan platform demonstrates the potential of combining external threat intelligence services with automated code analysis to improve web application security. The project highlights how such tools can support developers, students, and organizations in identifying security risks early and building more secure software systems. Below is the working of secure scan:

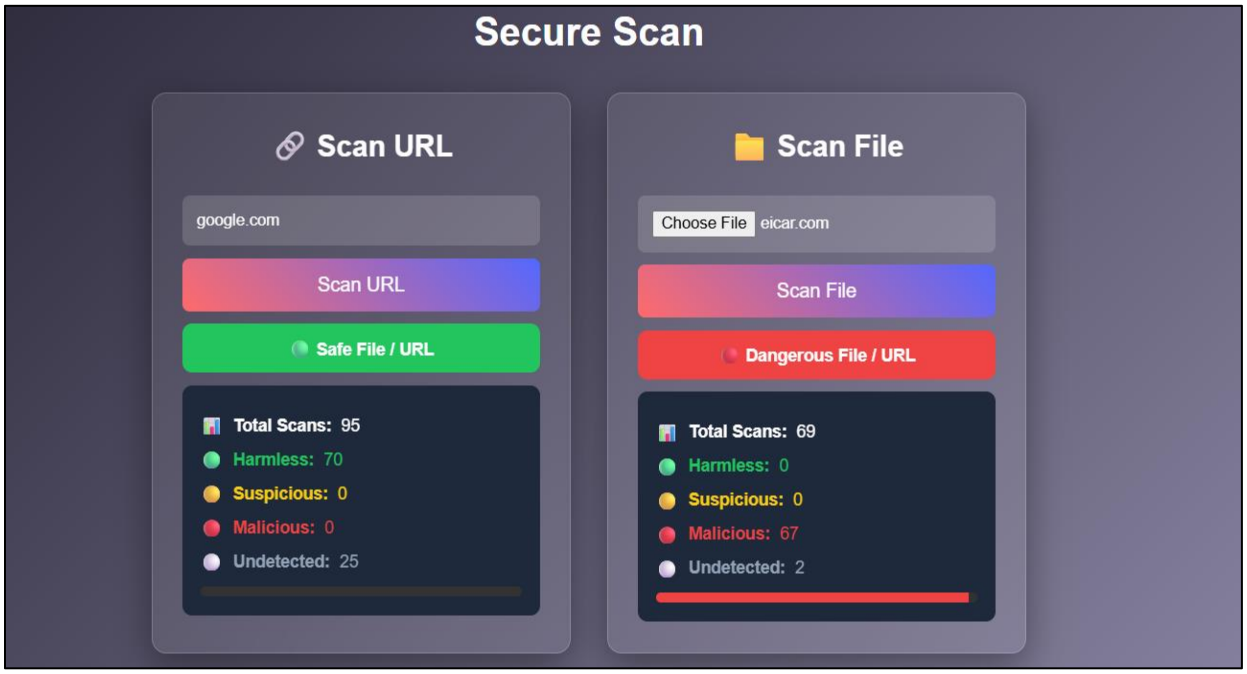


```
        <form method="POST" action="/login">
          <input type="text" name="username" placeholder="Username" /><br/>
          <input type="password" name="password" placeholder="Password" /><br/>
          <button type="submit">Login</button>
        </form>
      `);
});

app.listen(3000, () => console.log('Server running on http://localhost:3000'));
```

SQL Injection (SQLi): ❌ Potential SQL Injection detected! Use prepared statements.
Broken Authentication: ❌ Broken Authentication detected! Avoid hardcoded credentials and use strong authentication.
Sensitive Data Exposure: ❌ Sensitive Data Exposure detected! Never store sensitive data in plain text.

Fig 2 Secure Scan Code



Total Scans: 95
Harmless: 70
Suspicious: 0
Malicious: 0
Undetected: 25

Total Scans: 69
Harmless: 0
Suspicious: 0
Malicious: 67
Undetected: 2

Fig 3 Secure Scan

VI. CONCLUSION

In conclusion, the Secure Scan platform demonstrates an effective approach to improving web application security by combining multiple security analysis techniques within a single system. The project integrates malware detection and source code vulnerability scanning to help users identify potential cybersecurity threats during the development and testing stages. By analyzing files, URLs, and application code, the platform provides developers with useful insights into possible security weaknesses that may exist in their systems.

VII. FUTURE SCOPE

➢ *Expansion Beyond OWASP Top 10*

While the current platform focuses on the OWASP Top 10, future iterations could expand its scope to include other critical security frameworks and standards.

➢ *Integration with Emerging Technologies*

The rapid adoption of emerging technologies presents new challenges and opportunities for web application security. Future research could explore integrating the Secure Scan platform with:

- Internet of Things (IoT): As IoT devices become more prevalent, extending the platform to monitor and secure IoT-based web interfaces could mitigate risks associated with connected ecosystems.
- Blockchain-Based Threat Intelligence: Leveraging blockchain for immutable logging and decentralized threat intelligence sharing could enhance data integrity and collaboration among organizations.

➢ *Support for Cloud-Native and Serverless Architectures*

As organizations increasingly adopt cloud-native and serverless architectures, the platform could be extended to address their unique s;

- Container Security: Adding support for scanning Docker containers and Kubernetes clusters for misconfigurations and vulnerabilities would enhance protection for microservices-based applications.
- Serverless Function Monitoring: Monitoring AWS Lambda, Azure Functions, or Google Cloud Functions for insecure configurations and runtime vulnerabilities would ensure comprehensive coverage for modern web applications.

REFERENCES

[1]. OWASP Foundation, "OWASP Top Ten Project," 2021. [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed: Mar. 10, 2026].
[2]. Google, "VirusTotal – Analyze suspicious files and URLs to detect malware and automatically share them with the security community," 2024. [Online]. Available: https://www.virustotal.com/. [Accessed: Mar. 10, 2026].
[3]. A. K. Sharma and R. Gupta, "Automated Detection of Web Application Vulnerabilities Based on OWASP Top 10," International Journal of Information Security and Privacy, vol. 18, no. 2, pp. 45–60, 2023.
[4]. S. Patel, M. Desai, and P. Shah, "A Framework for Malware Detection Using Online Threat Intelligence Platforms," IEEE Access, vol. 11, pp. 98231–98245, 2023.
[5]. J. Brown and L. Carter, "Integrating Cyber Threat Intelligence into Secure Software Development Lifecycle," Journal of Cybersecurity Technology, vol. 7, no. 1, pp. 25–41, 2022.

[6]. R. Singh and V. Kumar, "Detection of SQL Injection and Cross-Site Scripting Attacks Using Static Code Analysis," Proceedings of the IEEE International Conference on Cyber Security and Protection of Digital Services, pp. 112–118, 2022.
[7]. Node.js Foundation, "Node.js Documentation: Event-Driven Architecture and Non-Blocking I/O," 2024. [Online]. Available: https://nodejs.org/. [Accessed: Mar. 10, 2026].
[8]. H. Lee and K. Park, "Real-Time Web Security Monitoring Using Node.js-Based Architecture," International Journal of Web Engineering and Technology, vol. 19, no. 3, pp. 210–224, 2024.
[9]. OWASP ZAP Project, "OWASP Zed Attack Proxy (ZAP) – Web Application Security Scanner," 2024. [Online]. Available: https://www.zaproxy.org/. [Accessed: Mar. 10, 2026].
[10]. PortSwigger Ltd., "Burp Suite – Web Vulnerability Scanner," 2024. [Online]. Available: https://portswigger.net/burp. [Accessed: Mar. 10, 2026].
[11]. M. Khan and S. Ali, "Cyber Threat Intelligence Platforms for Modern Web Applications: Challenges and Opportunities," Computers & Security, vol. 132, pp. 103–118, 2024.
[12]. P. Verma and N. Jain, "Static Code Analysis for Secure Software Development: A Survey," Journal of Network and Computer Applications, vol. 220, pp. 103600, 2023