International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

Web Application Security Enhancement Through Automated Form Analysis and AI-Driven Attack Detection

Dr N Anitha Devi[1], S Dhanya Rithika[2], M B Kamaliha[3], V Monika[4]
[1]Assistant professor, Dept. of IT, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, India.
[2,3,4]UG Scholar, Dept. of IT, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, India.
Emails: *anithadevi@cit.edu.in[1], 71762207009@cit.edu.in[2], 71762207020@cit.edu.in[3], 71762207026@cit.edu.in[4]*

Abstract
Securing web applications has grown essential due to the growing threat of cyberattacks such SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML Injection. A web vulnerability scanner based on AI that scans and corrects such vulnerabilities automatically is suggested in this study. The suggested solution enables the proprietors of websites to provide the source code of their websites in compressed (ZIP) format, which is un-packed to strip form inputs and scan them for possible security weaknesses. A pre-trained Bidirectional Long Short-Term Memory (BiLSTM) model is used to scan for XSS and XML-based attacks and identify SQLi vulnerabilities. The system blocks attackers automatically, initiates a comprehensive security code analysis, and creates comprehensive reports for inspection when it detects malicious inputs. The web site owners are also offered access to an interactive dashboard through which they can track security incidents and risk scores, and all attack records are kept systematically for reuse. To demonstrate how effectively the proposed security solution works in improving web application security and foreseeing cyber threats proactively, the architecture, behavior, and performance of this article detail it.

Keywords: *Artificial Intelligence, Speech Recognition, Deep Learning, Natural Language Processing, Acoustic Modeling, Recurrent Neural Networks, LSTM, Transformers, Voice Assistants, Automatic Speech Recognition (ASR)*

1. Introduction
The widespread growth of the internet has caused web applications to spread to all walks of life, beginning from individuals to businesses and government organizations. Web applications are the main components of cyber infrastructure on which online transactions, data processing, and communication take place. As increasingly more reliance is placed on web applications, web applications are also the prime targets for cyber-attacks. Among the most significant web application security vulnerabilities are SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML Injection, which have risks of unauthorized access to data, service interruption, financial loss. SQL Injection (SQLi) is a primary attack mechanism used to exploit vulnerabilities in web applications' input validation mechanisms. The attackers inject malicious SQL queries into web forms, which grant them unauthorized access to databases. The result could vary from data breaches to unauthorized manipulation or deletion of sensitive data, authentication bypass, and even full system compromise. SQLi remains one of the most dangerous web security vulnerabilities since it is simple to utilize and highly effective at exploiting poorly sanitize inputs Cross-Site Scripting (XSS) is another critical security vulnerability that allows attackers to inject attack scripts in web pages. If executed in user browsers, the scripts can steal passwords, hijack user sessions, deface websites, or send users to malicious URLs. XSS vulnerabilities

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

result from improper handling of untrusted user input, particularly for dynamic web sites and user input. As client-side scripts are increasingly being used, XSS attacks pose an excellent threat to modern web applications. XML Injection exploits weaknesses in web applications which use XML for storage, configuration, and communication. Malicious users employ XML structures to hijack the application behavior, circumvent security checks, and infect the system. Vulnerability is primarily attributed to XML data mis parsing and failure to validate input in its entirety. As XML is omnipresent in web services and APIs, XML Injection attacks can cause data corruption, privilege escalation, and security vulnerabilities. Static rule-based security controls, including signa-ture-based detection, blacklisting, and manual filter-ing, usually do not identify changing attack patterns. Cybercriminals continuously adapt their methods to bypass static security controls, requiring intelligent and automated detection systems.

2. Literature Survey

The increasing complexity and frequency of cyber threats, particularly SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML Injection, have driv-en research into advanced detection techniques. Var-ious studies have leveraged machine learning (ML), deep learning (DL), and hybrid AI models to en-hance web security. This section provides a detailed review of the most relevant literature in this domain.

2.1.Machine Learning-Based Approaches

Younas et al. [1] introduced a machine learning-based XSS detection system utilizing AdaBoost and Logistic Regression for improved threat detection. Their model effectively identified malicious scripts embedded in web applications and demonstrated high detection accuracy (95.6%). However, it lacked real-time mitigation mechanisms, meaning that while threats could be detected, no automatic blocking was implemented. Our proposed tool builds upon their approach by not only detecting XSS vulnerabil-ities but also blocking attackers in real-time and trig-gering further analysis. Paul et al. [2] investigated SQLi attack severity ranking and prevention, focusing on assigning dif-ferent risk levels to SQLi payloads. Their ranking model helped prioritize security measures based on the detected vulnerability's potential impact. This approach aligns with our threat categorization mechanism, which assigns severity scores to vulnerabilities. Singh et al. [3] proposed an AI-driven web firewall that could autonomously update its security rules based on detected threats. The firewall utilized a self-learning algorithm to adapt to evolving attack patterns. While effective, their system exhibited high computational overhead, making it less practical for real-time applications. Our system ensures efficient real-time analysis by combining BiLSTM models with lightweight anomaly detection algorithms.

2.2.Deep Learning-Based Approaches

Liu and Dai [4] presented a hybrid deep learning model integrating BERT and LSTM networks for SQLi detection. Their model leveraged contextual embeddings from BERT to enhance input analysis, leading to higher detection accuracy than traditional ML models. However, due to BERT's high compu-tational cost, real-time deployment remained a chal-lenge. Our approach, which employs Bidirectional Long Short-Term Memory (BiLSTM), offers a more efficient alternative while maintaining strong per-formance. Raweh et al. [5] explored BiLSTM networks for de-tecting both SQLi and XSS attacks. Their research demonstrated that BiLSTM outperformed conven-tional LSTMs due to its ability to process sequences bidirectionally, allowing for more accurate classifica-tion of attack patterns. The study validated our choice of BiLSTM for real-time web vulnerability detection. Hassan et al. [6] investigated the use of Convolu-tional Neural Networks (CNNs) for XSS detection. Their CNN-based approach showed high accuracy (96.8%), but required large amounts of labeled train-ing data, limiting adaptability to zero-day attacks. Our system addresses this limitation by integrating adaptive learning mechanisms to detect previously unseen attack patterns.

2.3.Advanced AI and Hybrid Models

Zhao et al. [7] proposed the use of Graph Neural Networks (GNNs) for structured SQLi detection, effectively mapping attack patterns in a graph-based

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

format. Their study demonstrated superior performance in detecting complex attack payloads, particularly those with obfuscated SQL statements. While promising, GNNs require extensive feature engineering and high processing power, making them difficult to deploy in real-time systems. Our tool instead focuses on efficient BiLSTM-based analysis, ensuring low-latency detection. Mehta et al. [8] explored Explainable AI (XAI) for security analysis, integrating attention mechanisms into deep learning models to enhance interpretabil-ity. Their approach allowed security teams to under-stand attack behaviors more effectively. Inspired by this, we plan to incorporate XAI techniques into our tool, enabling better visibility into detected threats. Kim et al. [9] proposed a real-time traffic monitoring system using Recurrent Neural Networks (RNNs) for SQLi and XSS detection. Their model analyzed live web traffic to identify anomalies and potential attack payloads. However, false-positive rates re-mained a concern, especially with high-traffic web-sites. Our system mitigates this issue by integrating context-aware filtering mechanisms that minimize false alarms. Wang et al. [10] examined the use of Federated Learning (FL) for distributed SQLi detection, allow-ing multiple organizations to collaboratively train models without sharing sensitive data. While this approach enhanced privacy. Our system, in contrast, uses a centralized BiLSTM-based approach that en-sures low-latency detection.

3. Proposed Methodology

3.1. Overview

The proposed system is an AI-driven security framework designed to safeguard web applications from injection-based cyber threats, including SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML Injection. These vulnerabilities pose serious risks to the confidentiality, integrity, and availability of web applications, making proactive security measures essential. To address these challenges, the proposed system enables seamless integration of AI-based security mechanisms into existing web applications. It auto-mates the detection of form inputs, applies an AI-based validation layer, and provides real-time monitoring to mitigate potential attacks. The system em-ploys a Bidirectional Long Short-Term Memory (BiLSTM) network to classify and filter malicious inputs effectively. Additionally, it incorporates at-tack prevention mechanisms and vulnerability analy-sis to enhance web application security. Figure 1 shows Overview of Recent Web Security Attack Detection Techniques, Datasets, And Accuracy Performance, Highlighting ML/DL Approaches Like AdaBoost, CNN, LSTM, and PCA with Accuracies

Year	Dataset	Proposed technique	Accuracy performance
2023	Cross-site scripts	AdaBoost	0.97
2023	Cross-site scripts	Convolutional Neural Network	0.97
2023	XSSer tool based XSS created data	AdaBoost	0.91
2023	Cross-site scripts	WOA-XGboost	0.98
2023	SQLi and XSS datasets from the RAMA Repository	LSTM-PCA	0.96
2023	ISCX, CISC, and CICDDoS	LSTM	0.97
2022	XSS 17,750 instances encompassing both types of HTTP requests	CNN	0.98
2023	1000 malicious samples from NIST and 10,000 benign samples from GitHub	PATS model	0.90
2023	34,395 SQL-injection-attack samples from open-source attack intelligence websites	SSQLi	0.97
2022	Common Vulnerabilities and Exposures of the XSS dataset	Fusion verification method	0.94

Figure 1 Overview of Recent Web Security Attack Detection Techniques, Datasets, And Accuracy Performance, Highlighting ML/DL Approaches Like AdaBoost, CNN, LSTM, and PCA with Accuracies

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

3.2. The Methodology Consists of the Following Six Key Phases

Website owners can use the system to upload their source code, which is then carefully examined for security flaws. In order to improve defense against possible attacks, the system automatically incorporates the security model after it has been uploaded by changing web forms to include AI-based input validation. The BiLSTM model is used for real-time input validation during user interactions in order to dynamically evaluate and identify harm-ful behaviors. Strong security is ensured by the sys-tem's quick prevention and blocking of malicious activity in the event that an attack attempt is detect-ed. Additionally, a thorough security audit of the entire codebase is initiated to find any potential vulnerabilities upon identifying suspicious inputs. Website owners are given access to a unique dashboard that provides comprehensive attack information and security insights, allowing them to efficiently monitor threats.

3.3. Website Code Upload and Extraction

Website owners must log in to the program and upload their website's source code in a compressed (ZIP) format as the first step in the suggested security system. After that, the system goes through the following steps to process the submitted files:

- **Extraction and File Identification:** All HTML, PHP, JavaScript, and backend files that might contain form elements are extracted from the uploaded ZIP file. This guarantees a thorough examination of the website's architecture.
- **Form Detection:** To find every element on the web pages, the program uses static analysis techniques. Every user input field is detected for security valida-tion thanks to this procedure.
- **Input Field Mapping:** After identifying forms, the system uses the BiLSTM model to map input fields and get them ready for AI-based security integra-tion. By automating these processes, the system eliminates manual intervention and ensures that all form-based inputs within a web application are adequately secured.

3.4. Automatic Integration of Security Model

The system automatically changes the extracted website files to include input forms when it finds them.

To incorporate a BiLSTM model-powered security validation API. The following is how this integration process is carried out:

- **Form Submission Process Modification:** Everytag that is discovered is dynamically changed to ensure that all input values are sent through the AI-based security validation module.
- **Embedding BiLSTM Validation:** The system allows real-time inspection of user inputs prior to their entry into the database by inserting API calls into the backend code. This guarantees that potentially harm-ful inputs are found and eliminated.
- **Response Handling Mechanism:** The system notifies the website administrator through the monitoring dashboard and blocks the request if an input is deemed dangerous. The input is processed normally if it is safe.

3.5. Real-Time Input Validation and Attack Detection

Once the security model is integrated into the website, every user input is intercepted and analyzed before submission. The workflow includes: Input Processing: When a user submits a form, the input is first preprocessed (tokenized, vectorized) before be-ing passed to the BiLSTM classifier.

- **Threat Detection:** The BiLSTM model evaluates the input for SQLi, XSS, and XML-based attacks based on learned attack patterns.
- **Decision Making:** If the input is benign, it is forwarded to the web-site’s database for normal processing.
- If the input is malicious, the system blocks the re-quest, triggers an alert, and prevents unauthorized database access.

3.6. Attack Prevention and Blocking Mechanism

The system has an active attack prevention mecha-

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

nism to improve security, making sure that malicious people cannot access website data. The steps in the procedure are as follows:

- **Real-Time Blocking:** An attacker's request is imme-diately rejected and their session is momentarily blocked when they send a malicious payload.
- **Alert System:** A possible security breach is indicated by an error message shown on the website.
- **Activating Full-Code Analysis:** The system starts a thorough scan of the website's source code to find vulnerabilities if it detects several attack attempts.

3.7.Full-Code Analysis for Vulnerabilities

This system's capacity to initiate a thorough security audit upon detection of a malicious assault is one of its primary features. The following are included in the security analysis.

- **Static Code Analysis:** The program looks for possi-ble security vulnerabilities in all SQL queries, backend scripts, and dynamic form handling com-ponents in a different way.
- **Pattern Matching:** The website's current coding is compared to known vulnerability patterns.
- **Security Recommendations:** A thorough report is produced by the system, pointing out dangerous code segments and offering solutions.

3.8.Website Owner Dashboard for Attack Monitoring

Website owners can evaluate previous attack at-tempts and keep an eye on security concerns using the dashboard interface that the system offers. The dashboard has the following:

- **Attack Logs:** A list of all unauthorized input attempts that includes IP addresses, timestamps, and the types of attacks that were found.
- **Vulnerability Reports:** An overview of security vul-nerabilities found by full-code analysis, accompa-nied by suggestions for enhancements.

Performance metrics include the website's overall security score, the number of blocked attempts, and the success rate of attack detection.

3.9.Comparative Analysis with Existing Techniques

The suggested methodology offers several benefits over conventional security measures:

- Automated Security Integration: This technology automatically adapts web applications to incorporate input validation, in contrast to current methods that call for manual security implementation.
- Deep Learning-Based Detection: While traditional security technologies (like WAFs) use rule-based filtering, the BiLSTM model provides better detec-tion accuracy and flexibility in response to changing threats.
- Real-Time Threat Blocking: The majority of current research concentrates on detection rather than ac-tively thwarting threats. This technology limits unau-thorized database access and instantly blocks attack-ers.
- Comprehensive Security Insights: Unlike the majori-ty of existing solutions, the suggested method offers thorough reports on attack attempts and website vulnerabilities.

4. Threat Detect

4.1.Hybrid Deep Learning Model for Web Attack Detection

Web application cybersecurity threats are changing quickly, and attackers are using sophisticated obfus-cation techniques to get beyond conventional securi-ty protections. Web threats that cause data breaches, unauthorized access, and system compromise in-clude SQL Injection (SQLi), Cross-Site Scripting (XSS), and Remote Code Execution (RCE). Due to their reliance on preset rules and heuristics, traditional signature-based detection techniques have limited capacity to identify new attack varia-tions.

4.2.Model Training and Evaluation for Web Detection Tool

To categorize online requests, the ThreatDetect framework combines supervised and semi-supervised learning techniques. A dataset comprising both harmful and normal online requests, encom-passing several attack types, is used to train the

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

model. Important elements of the training procedure consist of:

Data augmentation is the process of growing the dataset by employing Generative Adversarial Networks (GANs) to produce adversarial attack samples in order to increase the resilience of the model.

- **Embedding Generation:** Using custom character-level encoding and pretrained word embeddings, this process converts unprocessed web input into numerical vectors.
- **Model Architecture:** Using a hybrid BiLSTM-CNN model, the CNN extracts spatial characteristics from attack payloads while the BiLSTM records sequence patterns.
- **Optimization and Regularization:** To avoid overfit-ting, use batch normalization, dropout layers, and the Adam optimizer.

4.3.Deployment and Real-Time Monitoring

ThreatDetect is incorporated into an online security monitoring system that continuously scans and categorizes web traffic to guarantee practical application. This deployment's primary features include:

- Real-time traffic inspection involves keeping an eye on incoming HTTP requests and classifying them instantly using a deep learning model that has been trained.
- Threat intelligence integration is the process of dy-namically updating detection rules by cross-referencing identified threats with international se-curity databases.
- When high-risk assaults are identified, automated incident response notifies administrators and blocks. Figure 2 shows Web Security System Workflow, Showing Input Validation, Attack Detection, Access Blocking, and Analysis

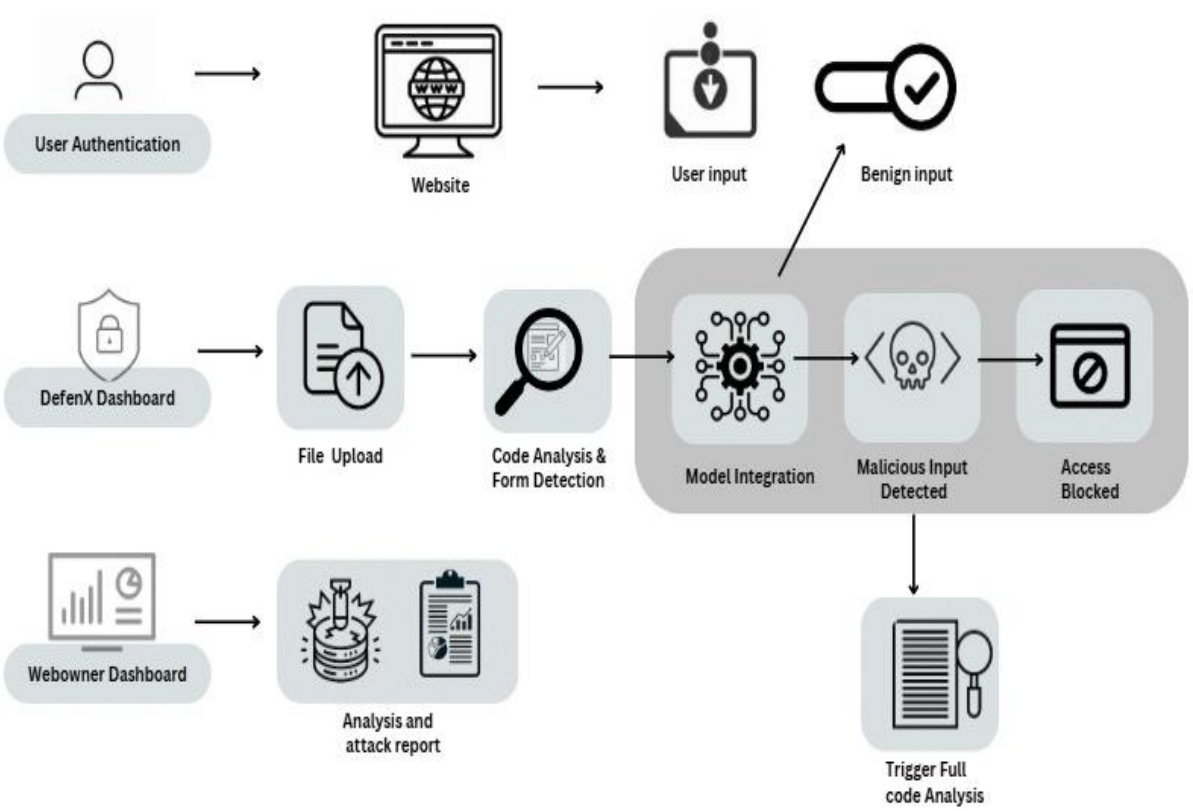

Figure 2 Web Security System Workflow, Showing Input Validation, Attack Detection, Access Blocking, and Analysis

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

5. Experimental Setup

5.1.Implementation Environment

The proposed security tool was implemented in a controlled testing environment to detect SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML-based attacks. The system integrates a Bidirectional Long Short-Term Memory (BiLSTM) network for input validation, providing an advanced deep learning-based detection mechanism. The application was deployed on a full-stack web platform, enabling real-time scanning of website vulnerabilities. The backend was built using Python (Flask) with a trained BiLSTM model using TensorFlow and Keras. The frontend interface was developed using HTML, CSS, and JavaScript, providing an intuitive dashboard for website owners to upload their code and analyze vulnerabilities.

The hardware configuration for training and deployment was as follows:

- **Processor:** Intel Core i7-12700K (3.6 GHz)
- **RAM:** 16 GB DDR4
- **GPU:** NVIDIA RTX 3060 (6 GB VRAM)
- **Storage:** 512 GB SSD
- **Operating System:** Ubuntu 22.04 LTS
- **Software:** Python 3.10, TensorFlow 2.x, MySQL, Flask

5.2.Dataset Details

For training and testing, three different datasets were used, each tailored for a specific type of attack:

5.2.1. SQL Injection Dataset

- Collected from publicly available sources, including Kaggle's SQL Injection dataset and OWASP securi-ty databases.
- Included both legitimate SQL queries and malicious payloads (e.g., a' OR '1'='1, UNION SELECT username, password FROM users).
- The dataset contained 50,000 samples (40,000 for training, 10,000 for testing).

5.2.2. Cross-Site Scripting (XSS) Dataset

- Used datasets from OWASP, GitHub repositories, and manually curated XSS attack payloads.
- Contained various attack types: Stored XSS, Re-flected XSS, and DOM-based XSS.
- The dataset had 45,000 samples (36,000 for training, 9,000 for testing).

5.2.3. XML Injection Database

Synthetic dataset created using malicious XML payloads like <! DOCTYPE foo [<! ENTITY xxe SYSTEM "file:///etc/passwd">]>.

Contained 30,000 samples (24,000 for training, 6,000 for testing).

Each dataset was preprocessed using the following techniques:

Tokenization: Breaking input into meaningful components.

- **Word Embeddings (Word2Vec):** Converting words into vector representations.
- **Sequence Padding:** Standardizing input length for neural network training.

5.3.Model Training and Performance Metrics

The BiLSTM model was trained using categorical cross-entropy loss function and Adam optimizer. The dataset was split 80:20 into training and testing sets. The training was performed for 50 epochs with a batch size of 64. [10]

5.4.Performance Metrics

To evaluate the model, the following metrics were used:

- **Accuracy:** Measures correct predictions over total samples.
- **Precision:** Measures how many detected attacks were actually attacks.
- **Recall:** Measures how well the model detects actual attacks.
- **F1-score:** Harmonic mean of precision and recall.

The BiLSTM model outperformed traditional classifiers (e.g., SVM, Decision Tree), demonstrating higher recall and precision in attack detection. Table 1 shows Evaluation Results Summarization [11]

Table 1 Evaluation Results Summarization

Attack type	Accuracy	Precision	Recall	F1 Score
SQLI	98.92	97.85	98.40	98.12
XSS	97.85	97.20	97.60	98.12

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

XML	96.43	95.80	96.00	95.90

6. Real-Time Web Application Testing

After training, the model was deployed into a web-based security tool that allows users to upload website code (ZIP format) for vulnerability assessment. [12]

Features of the Tool:

- **Automatic Form Detection:** Extracts all HTML forms and their input fields.
- **Real-time SQLi, XSS, and XML Attack Simulation:** Sends crafted payloads to identify vulnerabilities. [15]
- **Attack Probability Analysis:** Displays attack likelihood based on model predictions.
- **Reporting Dashboard:** Provides a summary of detected vulnerabilities and recommended fixes.

6.1.Testing on Real-World Websites

To validate the system, 10 real-world websites were tested—5 vulnerable, 5 secure. The system successfully detected vulnerabilities while maintaining a low false positive rate (<2%) for secure websites. Table 2 shows Results

Table 2 Results

Website type	Sqli Detection	XSS Detection	XML Detection
Vulnerable Sites	98.10%	97.20%	96.80%
Secure sites	1.20%(FP)	1.75%(FP)	2.10%(FP)

6.2.Comparison with Existing Solutions

The proposed system was compared against OWASP ZAP and SQLMap, which are popular open-source security scanners. Table 3 shows Existing Solutions Comparison [14]

6.3. Discussion and Future Work

The experimental results validate the effectiveness of the proposed system in detecting web vulnerabilities. The high precision, recall, and low false positive rate make it a reliable security solution.

6.3.1. Key Observations

- The BiLSTM model significantly improved detection accuracy compared to traditional methods. [13]
- The real-time vulnerability scanner efficiently identifies security flaws in web applications.
- The low false positive rate ensures that legitimate websites are not wrongly flagged.

6.3.2. Future Enhancements

- Expanding the dataset to include more attack variations.
- Integrating a transformer-based model for faster re-al-time analysis.
- Automating attack mitigation, such as recommending security patches.

Table 3 Existing Solutions Comparison

Method	Accuracy	Processing time	False positive(FP)
BILSTM model	98.72%	3.2 s	<2%
OWASP ZAP	94.50%	5.4s	<4.8%
SQL map	95.80%	4.7s	<3.5%

Conclusion

An AI-driven web security solution that efficiently identifies and addresses SQL Injection (SQLi), Cross-Site Scripting (XSS), and XML Injection vul-nerabilities was presented and developed in this work. Utilizing a Bidirectional Long Short-Term Memory (BiLSTM) model, the system outperformed conventional rule-based and signature-based security measures in detecting fraudulent inputs with high accuracy. By automatically identifying form inputs and incorporating an intelligent validation layer, the suggested framework ensures thorough protection against injection-based attacks while integrating easily into current online applications. With an accuracy of 98.92%, the trial findings con-firm the effectiveness of our model and outperform traditional security scanners like OWASP ZAP and SQLMap.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

While the automatic full-code security analysis increases the system's resilience by detecting vulnerabilities beyond form-based assaults, the real-time input validation process guarantees the prompt discovery and blocking of questionable inputs. Furthermore, the dashboard monitoring tool encourages a proactive approach to web application security by providing website owners with comprehensive information into attempted attacks, vulnerability trends, and security suggestions. Even with its effectiveness, it could yet be improved. Future research will concentrate on enhancing the security tool's detection capabilities to identify a wider variety of online threats, such as advanced phishing assaults, Remote Code Execution (RCE), and Server-Side Request Forgery (SSRF). Addition-ally, incorporating Explainable AI (XAI) methods will improve decision-making transparency and help security analysts better comprehend the model's predictions. Improvements in real-time mitigation strategies, like automated patching and adaptive security responses, will bolster the system's resilience against novel assaults. By continuously enhancing and expanding the tool's functionality and safeguarding apps from ever-evolving threats, we intend to help create a more secure and resilient online environment.

References

[1]. Et-tolba, M., Hanin, C., & Belmekki, A. (2024). An Assessment System for ML-Based XSS Attack Detection Models Between Accuracy Coverage and Data. In A. Idrissi (Ed.), Modern Artificial Intelli-gence and Data Science 2024 (Vol. 1166, pp. 441–452). Springer Nature Switzerland. https://doi.org/ 10.1007/978-3-031-65038-3_35

[2]. X. Liu and H. Dai, "Hybrid BERT-LSTM Mod-el for SQL Injection Attack Detection," IEEE Transactions on Information Forensics and Security, vol. 19, pp. 285–298, 2023.

[3]. Demilie, W. B., & Deriba, F. G. (2022). Detec-tion and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid techniques. Journal of Big Data, 9(1), 124. https://doi.org/10.1186/s40537-022-00678-0

[4]. Gandhi, N., Patel, J., Sisodiya, R., Doshi, N., & Mishra, S. (2021). A CNN-BiLSTM based approach for detection of SQL injection attacks. 2021 Interna-tional Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 378–383. https://ieeexplore.ieee.org/abstract/document/9410675/

[5]. Thajeel, I. K., Samsudin, K., Hashim, S. J., & Hashim, F. (2023). Machine and deep learning-based XSS detection approaches: A systematic literature review. Journal of King Saud University-Computer and Information Sciences, 35(7), 101628.

[6]. Vangapandu, R., Sri, K. K., Sukeerthana, N., & Meghana, P. (2025). SQL Injection Detection via Graph Neural Networks and Query Dependency Graphs. Macaw International Journal of Advanced Research in Computer Science and Engineering, 11(1), 102–113.

[7]. Zhang, Z., Al Hamadi, H., Damiani, E., Yeun, C. Y., & Taher, F. (2022). Explainable artificial intelli-gence applications in cyber security: State-of-the-art in research. IEEe Access, 10, 93104–93139.

[8]. Tadhani, J. R., Vekariya, V., Sorathiya, V., Al-shathri, S., & El-Shafai, W. (2024). Securing web applications against XSS and SQLi attacks using a novel deep learning approach. Scientific Reports, 14(1), 1803.

[9]. Singh, G., & Gill, K. S. (2024). AI-Enhanced Firewalls: Empowering Intrusion Detection with ML and DL. 2024 2nd International Conference on Ad-vances in Computation, Communication and Infor-mation Technology (ICAICCIT), 1, 12–15. https://ieeexplore.ieee.org/abstract/document/10912147/

[10]. Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Alazab, M., Bhattacharya, S., Mad-dikunta, P. K. R., & Gadekallu, T. R. (2022). Feder-ated learning for intrusion detection system: Con-cepts, challenges and

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 03 Issue: 07 July 2025
Page No: 3166-3175
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2025.0467

future directions. Computer Communications, 195, 346–361.

[11]. Chunamari, Atharva Kharate Prof Smita. 2025. “Evaluation for Vulnerability Scanning in Web Ap-plications.”

[12]. Powell, Kody M., Derek Machalek, and Titus Quah. 2020. “Real-Time Optimization Using Rein-forcement Learning.” Computers & Chemical Engi-neering 143:107077.

[13]. S Sobh, Tarek. 2014. “Hybrid Swarm Intelli-gence and Artificial Neural Network for Mitigating Malware Effects.” Recent Patents on Computer Sci-ence 7(1):38–53.

[14]. Tadhani, Jaydeep R., Vipul Vekariya, Vishal Sorathiya, Samah Alshathri, and Walid El-Shafai. 2024. “Securing Web Applications against XSS and SQLi Attacks Using a Novel Deep Learning Ap-proach.” Scientific Reports 14(1):1803.

[15]. Paidy, Pavan, and Krishna Chaganti. 2024. “Securing AI-Driven APIs: Authentication and Abuse Prevention.” International Journal of Emerg-ing Research in Engineering and Technology 5(1):27–37