

International Journal of
Engineering Research and Science & Technology



ISSN:2319-5991

www.ijerst.org

E-mail: editor@ijerst.org or ijerst.editor@gmail.com

MACHINE LEARNING-BASED SOLUTIONS FOR PROACTIVE WEB VULNERABILITY ASSESSMENT

¹Praveen,²Jeevan,³Swathi

¹²³Students

Department of CSE

ABSTRACT:

Web applications are increasingly targeted by cyber attackers exploiting vulnerabilities to compromise security, privacy, and data integrity. Traditional vulnerability detection techniques often fall short in terms of accuracy, speed, and adaptability to evolving threats. This paper explores machine learning-based solutions to enhance proactive web vulnerability assessment. By analyzing historical vulnerability data, source code patterns, and web traffic features, machine learning models can predict and detect vulnerabilities more effectively. The study reviews various supervised and unsupervised learning algorithms applied in this domain, evaluates their performance, and discusses the challenges and future prospects. Results indicate that integrating machine learning significantly improves detection rates and reduces false positives, thus enabling timely mitigation of security risks in web applications.

I. INTRODUCTION

The rapid growth of web applications in business, government, and social sectors has escalated the importance of securing these platforms against cyber threats. Web vulnerabilities such as SQL injection, cross-site scripting (XSS), and remote code execution are common attack vectors exploited by malicious actors. Conventional vulnerability detection methods, including static and dynamic code analysis, signature-based scanning, and manual penetration testing, often struggle with limitations such as high false positive rates, inability to detect zero-day vulnerabilities, and resource intensiveness.

Machine learning (ML) has emerged as a promising approach to overcome these challenges by enabling automated, adaptive, and data-driven detection of vulnerabilities. ML models can learn from large datasets encompassing past vulnerability reports, application logs, and network traffic to identify subtle patterns and anomalies indicative of potential security breaches. This paper focuses on machine learning-based techniques designed to proactively assess web application vulnerabilities, highlighting their methodologies, strengths, and weaknesses.

The objectives include:

- Investigating how various ML algorithms can be applied to web vulnerability detection,
- Evaluating the effectiveness of ML in improving detection accuracy and speed,
- Identifying current research gaps and potential directions for future work in this rapidly evolving field.

II. LITERATURE SURVEY

Several studies have demonstrated the potential of machine learning in enhancing web vulnerability detection:

- **Static Code Analysis with ML:** Wang et al. (2020) proposed a static code vulnerability detection method using a Random Forest classifier trained on syntactic and semantic features of source code, achieving high precision in identifying SQL injection flaws.
- **Dynamic Behavior-Based Detection:** Chen and Zhao (2019) utilized recurrent neural networks (RNNs) to analyze web

traffic patterns and detect anomalies associated with cross-site scripting attacks, showing improved detection rates over traditional rule-based systems.

- **Hybrid Approaches:** Li et al. (2021) combined static and dynamic analysis features within a support vector machine (SVM) framework to detect multiple vulnerability types, reducing false positives significantly.
- **Deep Learning Models:** Recent research by Kumar et al. (2022) employed convolutional neural networks (CNNs) on application log data to detect unusual access patterns indicative of exploitation attempts, outperforming conventional machine learning classifiers.
- **Challenges and Limitations:** Despite successes, ML-based methods face issues including imbalanced datasets, the need for extensive labeled data, vulnerability of models to adversarial attacks, and the difficulty of interpreting complex models in security contexts (Patel et al., 2021).

The literature emphasizes the need for robust datasets, hybrid techniques, and explainable AI to fully realize the benefits of ML in web vulnerability assessment.

III. WEB APPLICATION VULNERABILITY DETECTION METHOD

3.1. feature analysis of web application vulnerability categories based on machine learning

Web application vulnerability detection based on machine learning usually adopts two strategies: one is host based dynamic policy model security scanning, which checks the objects in the model that conflict with security rules, such as improperly set and vulnerable passwords; the

other is active policy scanning based on network security, which detects vulnerabilities and simulates attacks on the model by executing some components[3]. To Record the response of the model, and in order to ensure the effectiveness of vulnerability detection, combined with the principle of machine learning, the type characteristics of vulnerabilities are mined out, and the embedding coefficient of vulnerability data is used as the physical permission of network layer. The longer the model runs, the smaller the amount of permission is. In order to facilitate the compilation and application of subsequent functions, y_i represents the embedding coefficient of vulnerability data, and i represents the total number of network vulnerability data after obtaining the value[4]. Among them, the expansion quantity p of information behavior directly affects the physical load capacity of the network piece to the vulnerability. Like the coefficient attribute of the internal vulnerability, it is also the number of physical attribute permissions detected in the model. Set the expansion amount of information behavior to w , the parameter of data header node to q_i , and the average value of construction coefficient to d . the big data detection function of the model is as follows:

$$S = \int_{i=1}^e L \sqrt{\frac{2i}{q_i - 1}} \left(\frac{w_i}{y_i} \right) \cdot g |p| \quad (1)$$

In the above algorithm, L is the average detection constant of network chip vulnerability data, and g is the detection weight of vulnerability molecules. C + language is used to describe the vulnerability characteristics and calculate its syntax structure. Assuming that K pieces of information are mixed vulnerabilities in the encrypted network, the rules for detecting mixed vulnerabilities are obtained by declaring the rules of T vulnerability information. The function mode is as follows:

$$x_i = \max AT(S, K) \quad (2)$$

Where x_i is the function expression of vulnerability detection rules, S is the encrypted information of vulnerability, P is the detection parameter, and A is the accuracy of network information encryption when the vulnerability is invaded[5-7]. In order to prevent the effect of vulnerability detection is not ideal due to too many rules, before calculating vulnerabilities, the eigenvalues of each vulnerability are standardized to ensure that the mixed vulnerability noise to be detected and calculated is in a reasonable range. The following noise reduction calculation based on weakness:

$$f_i = \frac{Sx_i - 1}{\max(x_i) - \min(x_i)} \quad (3)$$

In Web application vulnerability detection rules, f_i represents a hybrid vulnerability detection rule, while $\max(x_i)$ and $\min(x_i)$ represent the maximum and minimum vulnerability characteristics respectively. These are the rules for detecting mixed attacks. In the process of encryption, real-time monitoring of mixed vulnerabilities is completed according to the encryption rules, which is prepared for the next step of vulnerability detection and calculation. Based on this, the HTTP protocol is constructed to divide the common vulnerability categories of applications. The HTTP protocol itself is stateless, that is, there is no dependency between different web requests. A network application vulnerability detection model is developed. Its main function is to detect the vulnerability of the target site. The detection model automatically mines cross site script vulnerabilities in the site, and stores the vulnerability information found in the vulnerability information database for users to view. First, submit the test response on the website, and then submit the vulnerability detection and response points on the website. Based on the response information to determine

whether there is a web vulnerability. After crawling the seed URL, all the web links on the seed URL are synchronized by multithreading, which improves the efficiency of crawling model. Using a variety of vulnerability detection tools can shorten the working time of vulnerability detection module several times, and greatly improve the efficiency of network vulnerability detection. Based on this, the vulnerability location model of web application is optimized, which is shown in Fig. 1.:

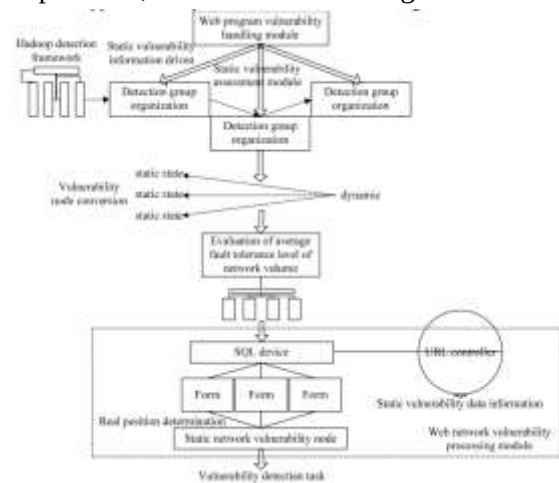


Fig. 1 web application vulnerability location model

In this model, vulnerability detection model is divided into three functional modules: network vulnerability detection function, test case generation function, vulnerability detection function. Some people think that collecting information and submitting data of some pages can only succeed when the user logs in, while the form of some pages contains verification code, so the verification code must be submitted when submitting information. The simulation login function and verification code identification function are added. In network application, a security vulnerability detection method based on agent is proposed.

3.2. Implementation of web application vulnerability detection

According to the above definition of vulnerability detection algorithm, in the process of network information encryption, mixed vulnerabilities can be accurately detected, but it is also necessary to accurately locate the vulnerabilities. In the process of network information encryption, the current location of the vulnerability is updated by using the cluster relationship, and the current location is recaptured until the exact location of the mixed vulnerability is found. When the mixed vulnerability is found, the tracking process in X_j state must conform to the clustering relationship. At this time, any state X_i can be selected for iterative calculation in the tracking process

$$X = (X_i + f_i)Rand(j), X_i \in X_i \quad (4)$$

According to the above definition of vulnerability detection algorithm, the mixed vulnerability can be detected accurately when the network information is encrypted, but it is also necessary to accurately locate the vulnerability. This method uses the clustering relationship between PSO algorithms to update the current location of vulnerabilities and relocate them until the exact location of mixed vulnerabilities is found in the process of network information encryption. When the mixed vulnerability is found, the tracking process in X_j state must conform to the clustering relationship. At this time, the state X_i can be arbitrarily selected for iterative calculation in the tracking process.

$$P_f = \min C, f \in X \quad (5)$$

In the detection process, P_f represents any function. It is the weakness location coefficient; C is the minimum clustering range. Then, the vulnerability location coefficient is determined by the finite iteration method to find the best clustering location, that is, the mixed vulnerability location. The real-time detection and location of mixed attacks on network information encryption are realized. Further,

combining with machine learning principle, the steps of Web vulnerability detection are optimized. Firstly, the page information is obtained, and the injection point of interaction with users is obtained in the page, so that the user can submit the data position and other data in the form. Then, the HTTP request is submitted to the target site and submitted to the web server to analyze the response information of the server to determine whether there is a cross site scripting vulnerability, and its function is optimized, which is shown in Fig. 2.:

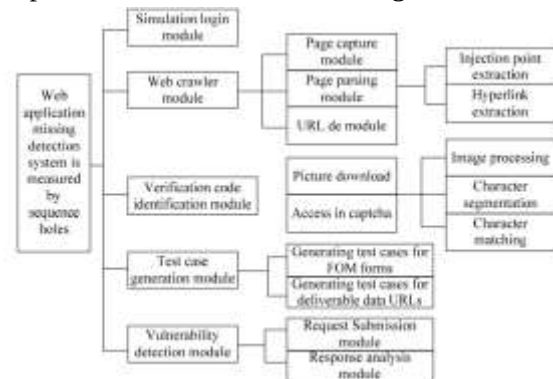


Fig. 2 function optimization of web application vulnerability detection model

This model mainly includes five functional modules: network vulnerability detection module, vulnerability detection module, simulation login module, authentication and identification module, test case generation module. The web vulnerability detection module collects and analyzes all the web pages of the target website, extracts the injection point information from the web pages that users can submit, and then inputs it into the model database. Based on this, the vulnerability detection steps are simplified, which is shown in Fig.3.:



Fig. 3 Web application vulnerability detection steps

In order to ensure the authenticity of the detection results of vulnerability components, it is necessary to collect and process the intrusion behavior parameters determined during the operation of the model before the model performs big data functions and other operations. In the absence of strong external interference, the vulnerability information is decomposed according to the determined detection direction, and a large number of connectable nodes are sent to the intermediate actuator. The request submission module submits the test cases, and the response analysis module analyzes the server response, so as to realize the accurate detection of web program vulnerabilities and ensure the security of program operation.

IV. ANALYSIS OF EXPERIMENTAL RESULTS

The test run environment is an Intel cor 3GHz p7450 CPU with 48 GB ram. Dacapov2006-10mr2 and SPECjvm98 benchmark program 2 are used in the experiment. The CPU of the experiment is Intel (R) core (TM) i5, and the computer memory is 4 GB. EVMWARE:78 Core processor; memory: 16 g; hard disk: 32 g; operation model: a-buntu 64 bit model (hhvm only supports 128 bit model); software environment: Windows 7 operation model, 64 bit operation model, 15 m network bandwidth, SQL Server 2008 database, jdk80 java development and MyEclipse development environment. The experimental code uses the source code, without running any code, which saves the trouble of creating network manager.

Based on the above detection environment, the experimental detection is carried out. Taking 10 minutes as a unit time length, the total amount of vulnerability detection of the traditional method and the method in this paper under five unit time

length are recorded respectively. Two models of the experimental group and the control group are used for detection. The test details is shown as Table 1.

Table 1 Comparison of total amount of vulnerability detection in unit time

group	Experiment time/ (min)	control group/ ($\times 10^4$ T)	experience group/ ($\times 10^4$ T)
1	5	2.4	4.4
	10	2.1	4.5
2	15	2.5	4.6
	20	1.9	4.7
3	25	2.6	4.8
	30	1.5	4.9
4	35	2.7	5.0
	40	1.3	5.1
5	45	2.8	5.2
	50	1.2	5.3

Based on the above detection results, compared with the traditional methods, the proposed web application vulnerability detection method based on machine learning in the same detection environment and detection time, the amount of vulnerability detection is significantly better than the traditional method. The error detection rate of the two methods was further compared and recorded. The specific detection results is shown as Fig.4:

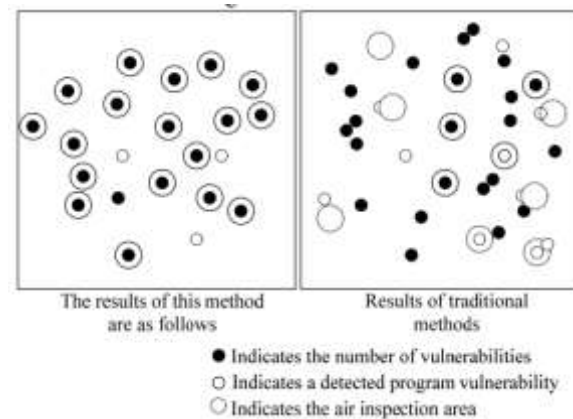


Fig. 4 Comparison and detection results of web application vulnerability false detection rate

Based on the above detection results, it is not difficult to find that in the same environment, the web application vulnerability detection method based on machine learning can detect vulnerabilities more accurately in the actual application process, and the number of false and missed detection is significantly reduced, which

fully meets the research requirements. Compared with the common vulnerability detection methods, the vulnerability detection module of this method has the following advantages: the login simulation function is added to ensure that the process of page vulnerability detection is carried out in the user login environment; it contains an effective URL duplicate data removal module, which can filter and delete the extracted duplicate hyperlinks and avoid vulnerability detection crawling duplicate pages. The extracted hyperlink is converted from a relative path to an absolute path to reduce the possibility of web page crawling failure. Multi thread technology is used to make multiple vulnerability detection programs crawl at the same time, which improves the work efficiency. In this way, at the same depth of crawling, the network vulnerability detection of this model can capture the web pages more effectively. At present, there are hidden dangers in the security, compatibility and performance of intrusion detection technology of web browser. The research of intrusion detection technology based on browser can not only effectively improve the overall security of browser, but also improve the compatibility with browser.

V. CONCLUSION

Machine learning-based solutions offer significant advancements in proactive web vulnerability assessment by enabling more accurate, faster, and adaptive detection capabilities compared to traditional methods. The ability of ML models to learn complex patterns from diverse datasets helps in identifying vulnerabilities that might otherwise remain undetected. However, challenges such as data scarcity, adversarial resistance, and interpretability remain critical areas for future research.

To maximize the effectiveness of ML in web security, future efforts should focus on

developing comprehensive datasets, enhancing model robustness against attacks, and improving transparency of decision-making processes. Combining ML with traditional security measures in a hybrid framework promises a balanced and effective defense strategy against the ever-evolving landscape of web vulnerabilities.

REFERENCE

1. Tan B , Elnaggar R , Fung J M , et al. (2020) Towards Hardware-Based IP Vulnerability Detection and Post-Deployment Patching in Systems-on-Chip[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, PP(99):1-1.
2. Yi M , Xu X , Xu L . (2019) An Intelligent Communication Warning Vulnerability Detection Algorithm Based on IoT Technology[J]. IEEE Access, 7(99):164803-164814.
3. Kurniawan A , Abbas B S , Trisetarso A , et al. (2018) Static Taint Analysis Traversal with Object Oriented Component for Web File Injection Vulnerability Pattern Detection[J]. Procedia Computer Science, 135(4):596-605.
4. Dharmalingam J M , Eswaran M . (2018) An Agent Based Intelligent Dynamic Vulnerability Analysis Framework for Critical SQLIA Attacks: Intelligent SQLIA Vulnerability Analyzer Agent[J]. International Journal of Intelligent Information Technologies,14(3):56-82.
5. Fu M , Wu L , Hong Z , et al. (2019) A Critical-Path-Coverage-Based Vulnerability Detection Method for Smart Contracts[J]. IEEE Access, PP(99):1-1.

6. Qin J , Zhang H , Guo J , et al. (2020) Vulnerability Detection on Android Apps–Inspired by Case Study on Vulnerability Related with Web Functions[J]. IEEE Access, PP(99):1-1.
7. Kaur A , Nayyar R . (2020) A Comparative Study of Static Code Analysis tools for Vulnerability Detection in C/C++ and JAVA Source Code[J]. Procedia Computer Science, 171(5):2023-2029.