International Journal On Engineering Technology and Sciences – IJETS
ISSN (P):2349-3968, ISSN (O): 2349-3976 Volume XIII - Issue III, March - 2026

AI-POWEREDWEBAPPLICATION VULNERABILITY SCANNER

[1]Miss. Sanjana, [2]Miss. HarshithaM, [3]Miss.Suraksha, [4]Miss.Sushmita [1]Assistant Professor, [2,3,4]UG Students,
Department of Cyber Security ACS College of Engineering Bengaluru-560074, Orcid: 0009-0003-3988-8326,
sanjanachavan98@gmail.com

Abstract — To cope with the growing number of evolving threats to web applications through the Internet, organizations must implement new technologies that allow for intelligent and adaptive security assessment of those applications. The traditional methods for scanning web applicationsutilizedrulesand staticsignaturesbasedonhow Viruses or Malware are detected by Anti-virus programs; however, those same techniques are not adequate to protect againstobfuscated(masked)orzero-daytypevulnerabilities. In this research paper, We present a web application vulnerability scanner (WA-VScan) that uses artificial intelligence (AI) technologies and employs an ensemble of anomaly detection techniques to perform accurate and timely identification of web application vulnerabilities. The WA-VScan uses a hybrid approach consisting of isolation forest and one-class SVM (support vector machine) algorithms to detect anomalous responses from the web application while providing contextual information to the security analyst via the integration of the Common Vulnerabilities and Exposures (CVE) and the OWASP Top Ten Vulnerabilities Reference Lists. The WA-VScan has been developed and implemented as a web application built on the Flask/PostgreSQL platform with the following features:useraccountsanduserauthentication,trackingof scan history, smart crawling, payload detection, creation of automated PDF reports. The test results indicated that the WAVScan had an overall accuracy of 83.3% for baseline validationusingstandardmachinelearning(ML)validation and was able to identify 100% of the previously identified vulnerable web applications tested with a much lower false positive rate than traditional techniques would have produced. The data obtained demonstrate WA-VScan's practical application and effectiveness in performing web application vulnerability assessment as a modern AI-driven tool.

Keywords — vulnerability scanner; web application security; Flask; ensemble learning; IsolationForest;One- ClassSVM; OWASPTop 10;CVE mapping; PDF report; cybersecurity automation.

I. INTRODUCTION

Web Applications are widely utilized across all industries, including E-Commerce and Digital Banking, which makes them attractive targets for cyber criminals [9], [22].ThecontinuedprevalenceofDatabaseFormatInjection, Cross-Site Scripting, and Command Injection attacks is a clear exampleof how vulnerable these applications are [12], [15],[16].Traditionaltoolsusedbyorganizationstoidentify and protect against these attacks, including Burp Suite and OWASPZAP,utilizeeitherarules-basedorsignature-based detection model [11], [19] and therefore are not able to identify Zero-day, Obfuscated, or Behavioural attacks. This results in a high false positive rate.

As more advanced attack vector exploitation strategies are developed by attackers, defensive measures must also adapt. Artificial Intelligence and Machine Learning offer uniquely compelling capabilities to identify patterns, detect abnormalactivities,and toidentify unknownthreats [2],[4], [22]. As such, an Artificial Intelligence-based web application vulnerability detection solution that incorporates Behavioural Analysis, Ensemble-based Anomaly Detection.

This solution leverages intelligence gathered from intelligent crawlers, automated payload testers, machine learning-based anomaly detection, as well as contextual intelligence gathered from public databases (Common Vulnerabilities and Exposures, or CVE) and the Open Web Application Security Project, and creates a real-time report detailing the vulnerabilities present [7], [8], [13] and supporting the secure development of web applications.

A. ProblemStatement

Conventionalwebvulnerabilityscannersareoftenslowto detect newly developed or obfuscated attacks and have high false-positiverates,whilefailingtodetectsomesophisticated attacks [19], [11].Therefore,thereisaneedfor an intelligent and adaptive system that analyzes web traffic behavioral patterns, detects anomalies in real time, and classifies vulnerabilities using AI-driven techniques.

B. Objectives

Our primary objective is to design and develop an AI-poweredwebvulnerabilityscannerthatiscapableoflearning about and adapting to changing behaviors related to attack patterns. The system has the aim of reducing false positives, improving overall detection accuracy, providing explainable threatscoring/detectionmechanisms,andprovidingadetailed report on vulnerabilities discovered throughout the process towards secure web application development and maintenance.

II. LITERATUREREVIEW

1. ArtificialIntelligenceandDynamicAnalysis-Based Web Application"[1]

This article proposes an AI-based vulnerability scanner that leverages dynamic runtime analysis to detect vulnerabilities in modern web apps. The authors instrument the web app to capture real-time behavioral data such as request-response behavior, execution control flow, latency differences, and content deviations. This logged behavioral data is preprocessed and fed into ml classifiers to automatically determine what is normal and what is anomalous behavior.

The advantage of this approach is that it enables the detection of runtime-only vulnerabilities which are not detected by static code analysis alone. This model is built off behavioral feature extraction, anomaly scoring based on thresholds, and temporal correlation analysis to reduce false positives. The model even adapts to never-before-seen attack patterns through model retraining and feedback learning.

Multiple web applications tested showed improved detectionprecisionandcallcomparedtotraditionalsignature-based vulnerability scanners. The authors point out that dynamic behavioral monitoring enables the use of ML anomaly detection to increase coverage and improve total scanning response time. Some limitations to the system still require discussion including: labeling required for training dataset, increased conflict due server request overhead in heavily trafficked environments, and generalization within heterogeneous environments. Detection [1].

2. "AI-Based WebVulnerabilityScanner:A ComprehensiveReview"[2]

This survey aggregates and reviews the current literature on AI-based web vulnerability scanners. It first presents a taxonomyofvulnerabilitytypes(e.g.,OWASPTop10)andthe existingcategoriesoftools.Itthen discussesthe potentialofAI models for anomaly detection, supervised classification, and hybrid methods. The authors also compare strengths and weaknesses—AI can be used to adapt and find zero-day threats, but data scarcity and interpretability are issues. The review also reveals that integrations with ongoing security processes are often overlooked. In conclusion, the authors present open problems, including continuous learning, model explainability, and reducing false-alarm rates [2].

3"AutomatedVulnerabilityAssessmentUsingMachine Learning"[4]

This document investigates the use of ML techniques to automateandimprovevulnerabilityassessments.Theauthors demonstratehowMLcanhelp,giventhesizeof the data, and understand models of vulnerabilities and patterns to make predictions with accuracy. The authors provide a framework in which a classifier consumes features from metadata from code, system logs, or network traces and provides prioritized candidates for the vulnerabilities. Integrated, it supports reducing human effort and increases the speed of detection. The authors discuss limitations such as the dependency on high-quality datasets, as well as overfitting. Their experimentation shows better accuracy compared to classical tools in some domains. Finally, the authors will stress that hybrid systems (ML) give the best tradeoffs[4].

4. "AI-BasedSoftwareVulnerabilityDetection:A SystematicLiteratureReview"[5]

This paper systematically reviews literature published between 2018 and 2023 concerning bridges of artificial intelligence and machine learning (AI/ML) approaches to software vulnerability identification.The authors offer a taxonomy of the methods in this literature, including feature extraction representation/embedding methods and deep learning methods or learning approaches. The authors provide some analysis of trends in the literature, including using deep learning almost becoming as dominant as traditional approaches and the increasing use of hybrid methodologies that involve both static analysis and dynamic analysis.They also discuss apparent gaps in the literature. Thesegapsinclude:(1)alackofastandardisedbenchmark, (2)alackoftransferabilityofmethodsbetweentheidentified cases, and (3) poor interpretability. Alongside their identification of gaps, the authors explicitly suggest future work/usesofhybridmodelsand adversarial robustness.The authors' review represents a strong road map [5].

5. "WebApplicationVulnerabilityDetectionMethod BasedonMachineLearning"[13]

The authors of this paper describe a machine learning method dedicated specifically to Cross-Site Scripting (XSS) detection in web applications. They review the existing techniques for detection, and then provide a model that has learned to classify input patterns in request parameters into benign and malicious classes. As Through our literature review, it was observed that AI/ML technologies have been incorporated for both static and dynamic vulnerability detection, with regards to XSS, CSRF, SQL injection, and othervulnerabilities.Generallyspeaking,hybridmodels(rule-based+learning-based)performbetter.Thekeychallengesare related to the limited availability of labeled datasets, overfitting to app(s), transferability of models to unseen systems,andtransparency/interpretabilityofmodeldecisions. Theproposedworkwilllooktoaddressthesimpedimentsby designing a modular feature extraction, cross-app generalization, continuous model updates, and explainable output modules. Part of their model, they also include CAPTCHA bypass simulation and a server filtering bypass. Theirexperimentsrevealrelativelylowerratesoffalsealarms and missed detections compared to baseline methods. They particularly highlight preprocessing steps and also use multi-threaded crawling for more effectiveness. Nevertheless, their model is limited in scope mainly to XSS detection, and generalizationtoothertypesofvulnerabilitiesmaybedifficult [13].

6. "Machine Learning for Web Vulnerability Detection: The Case of Cross-Site Request Forgery" [3]

This paper presents Mitch, a machine-learning-based framework for identifying CSRF vulnerabilities within web applications. The authors formulate the detection as a black-boxissue,relyingonfeaturesgatheredfromtracesofrequests and responses, header patterns, reusable tokens, and anomaly detection. The evaluation of Mitch is conducted within the context of real-world application benchmarks, indicating that theframeworkhastheabilitytoidentifyCSRFthreatswithan acceptablelevelofprecisionandrecall.Theauthorsarticulate some of the challenges including diverse application logic, falsepositivesdueto benign anomalies, and feature selection. In addition, the authors suggest methods to generalize across webapplicationswithnormalizeddistributionsofthefeatures. The contribution demonstrates the potential value of machine learning augmentation for even the protocol-based vulnerabilities such as CSRF [3].

A. Synthesisandgapsidentified:

Through our literature review, it was observed that AI/ML technologies have been incorporated for both static and dynamic vulnerability detection, with regards to XSS, CSRF, SQL injection, and other vulnerabilities. Generally speaking, hybrid models (rule-based + learning-based) perform better.

III. METHODOLOGY

A.SystemArchitecture

ThisisanexampleofalayeredandmodularArchitecture Which Utilizes an Artificial Intelligence Based Solution. It allows the collection of information from The Web, analysing the Collected Information to identify Security Gaps in The Web Application and Provide Insightson How To Fix The Issues Detected By The Scanner. The General Workflow of The System is Depicted in (Fig. 1).

1) TrafficCaptureandPreprocessing:Trafficprocesswillbe capturing HTTP requests and responses, HTTPS requests and responses, as well as all data that is generated between thetargetapplicationandtheenduserduringtheirbrowsing experiencewiththatwebapplication.Allofthisdatawillbe cleaned up and normalized for easier use in downstream feature extraction and analysis. By pre-processing the captured traffic, it allows us to create a structure that is consistentacrossalldatacaptured,sothatwecanextractthe necessary features from the data and analyze the data more efficiently [7], [8].

2) Feature Extraction: The system uses pre-processed software traffic to extract two distinct types of features: Statistical and Behavioural. The statistical features include thingslike:lengthofrequestparameters,entropy,frequency ofincomingrequestspersecond,attributesofanyheadersin the HTTP requests/responses, response status codes and patterns of access to the application[3], [13]. The behaviouralfeaturesdepictwhattype of interaction theuser has with the web application. The combined extracted featurevectorsthenserveasinputforthevariousmodelsthat thesystemusestoidentifythevulnerabilitiesinthesoftware, along with any other critical information related to these interactions.

3) Detection and Classification: The detection of vulnerabilities uses a Hybrid Machine Learning Approach[4],[24].Theprimarymethodofdetectionisbased upon distinguishing between normal and anomalous behaviourofwebtraffic.Onceanymaliciouswebtraffichas been identified, the system will then classify the specific vulnerability that has been detected, for example: SQL Injection or Cross-Site Scripting. By employing a multi-layered detection approach, the system can reduce the numberoffalsepositiveswhilealsomaintaininghighlevels of accuracy for vulnerability detection.

4) ExplainabilityandThreatScoring:Foreachvulnerability identified,athreatseverityscoreisgeneratedbythesystem, based on the detected impact of the vulnerability and the confidencethatthemodeldetectedthevulnerability[25].To provide insight into how the model reached the conclusion that a vulnerability exists, an explainability module outputs information about features contributing to the model's scoring decision, thus increasing transparency and analyst trust.

5) FeedbackandModelUpdate:Analystfeedbackisusedto refinethedetectionresults.Confirmeddetectionsareusedto continue to retrain and update the models as new patterns emerge.

6) StorageLayer: Thesystemusesatwo-databaseapproach for data storage efficiency: MongoDB stores unstructured data(e.g.,rawtrafficlogsandfeaturesets),andPostgreSQL stores structured data(e.g., results from scans, vulnerability metadata, and threat scores).

7) User Interface and Reporting: The system gives users an interactive user interface that presents vulnerabilities detected and their severity scores, along with visual summaries of the vulnerabilities. The system also generates detailed and exportable reports that security analysts use to assess vulnerabilities for remediation.

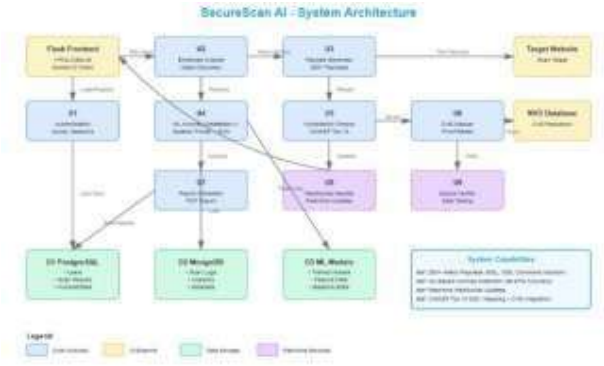


Fig.1.SystemArchitectureofAI-Powered VulnerabilityScanner

B. AlgorithmsUsed

Isolation Forests. Isolation Forests recursively splits the feature space by partitioning the different features of the data[22].Datapointsthattakethefewestnumberofsplitsto isolate are considered anomalies. Isolation Forests is best used with high-dimensional HTTP feature vectors, and it effectively detects rare and unusual behaviours such as abnormal parameter entropy, unusual request payloads, or anomalous server responses.

One-Class Support Vector Machine (OS-SVM). OC-SVM uses a model of normal traffic to classify deviations as anomalies[3]. OC-SVM is very effective at identifying subtletypesofattacks(e.g.,unusualheadermanipulationsto POSTrequestsorunusualpatternsofaccessthatmayevade traditional signature detection methods).

HybridDetectioncombinestheoutputofunsupervised anomaly detection systems with those generated by supervised classification systems trained on labelled vulnerablesets.Thehybriddetectionstrategyenhancesthe system's ability to identify previously unseen attacks as well as vulnerabilities known from signature databases. Consequently, the hybrid detection system is more robust and adaptable than traditional signature-based detection systems[4], [24].

IV. IMPLEMENTATION

A. OverviewofArchitecture

The Layered Architecture Model for WA-VScan is made up of four layers: Traffic Capture Layer, Feature Extraction Layer, Machine Learning Based Detection Layer, Reporting Module Layer. Each of these four components can operate independently and communicate via RESTful APIs which are implemented with a Flask Framework. Each Layer can be considered as an independent piece of software and therefore allows scalability. The WA-VScan layered architecture has the capability to produce automated scans and has Reporting Capability..

B. Setupand Environment

The WA-VScan back-end is built using the FLASK framework by using the Python 3.11 Programming Language[2]. The Hybrid Storage Model has three layers ofdatastorage.Thedatafromtheunstructureddata(HTTP Protocol) will be stored into the MongoDB Database. The structured data from the Digital Certificate Scan Results will be stored into the PostgreSQL Database. Request, BeautifulSoup,Scikit-learn,JoblibareusedasLibrariesfor the Core support of the Program.

C. TrafficCaptureandPreprocessing

AnAutomatedWebCrawlerModelwasusedtomimic a human being/robot, the interaction(s) with the Target WebsiteandcapturetheHTTP/SProtocolTrafficfromthe Web Server. Every HTTP Request/Response will store method of transport protocols (GET/POST/etc.), the URL of the request, the headers sent while making the request, sentpayload,theHTTPResponseStatuesfromtheServer, and the time taken (latency) for the request to process. Unprocessed logs will be stored into the MongoDB Database while the Summary Logs will be stored into the PostgreSQL Database.

D. FeatureExtractionandEngineering

Statistical and Behavioural Features are extracted from HTTP Protocol Traffic Capture, Examples of these features would include, Number of parameters; Average Number of Runtime Requests Per User; Number of Runtime User Interactions; Average User Interaction Time (seconds) for AllRunTimes;TotalNumberofUsersWhoHaveAccessed Target URL; etc.

E. ModelTrainingandValidation

Forexample,onetypeofMachineLearningalgorithmis calledUnsupervisedMachineLearning(UML)andincludes models such as Isolation Forest (ISO) and One-Class SupportVectorMachine(OCSVM).UMLwillhelpidentify patternsinyourtraffictohelpyoudetectvulnerabilities.You willtrainyourmodelswithyourdatabyusingtwodatasets, one for training and one for testing. You will evaluate how well your trained models perform by looking at the "Precision", "Recall", "F1 score", and "False Positive Rate" metrics for your model. Also, use Cross-Validation so that you know your trained model will work for other datasets.

F. InferenceandAlertGeneration

The trained machine learning models are deployed to a Flask-based inference engine, which is what allows us to take in requests and responses that we capture. For each request and response pair that is captured, the models score the features that we extract. If a request exceeds a defined anomalythreshold,therequestisflagged,andweprovidean associated severity score and include it in the logs for an analyst to review.

G. ReportingandEvaluation

We provide a web-based dashboard that provides informationondetectedvulnerabilities,theirseveritylevels, and the overall scanning process. Reports of detected vulnerabilitiescanalsobecreatedbyexportingthePDFand JSON. Testing of the model was conducted in both test environments, DVWA and OWASP Juice Shop, with a reduction in the false positive rate for anomaly detection as compared to traditional rule-based scanning approach[12], [19].

H. ImplementationLimitations

The current implementation cannot effectively detect vulnerabilities that result from complex business logic. Applicationsthathaveheavyclient-sidescriptingwilllikely take much longer to crawl through and evaluate for anomalies.Toensureeffectivedetectionofevolvingpatterns of attack, periodic retraining of the models is necessary.

V. RESULT&DISCUSSION

The AI-Powered Web Application Vulnerability Scanner (WA-VScan) was tested in three separate environments DVWA, OWASP JuiceShop, and Acunetix TestPHP VulnWeb to determine itseffectiveness, accuracy, and usability.

TheScanning Dashboard is the main user interface for beginning scans, tracking scanning progress, and reviewing summaryresultsinreal-time(Fig.2).Thedashboardallows userstoseethedifferentstagesofthescanningpipelineand shows users where they are in the scanning process, including Recon, Payload, ML analysis, and Report generation.

(Fig.3)istheaccountcreationandauthenticationmodule for first-time users. This module provides secure access controls for new users, enables them to create an account, and keeps their accounts secure through controlled user sessions and input validation. First-time users can therefore have a personalized history of scans performed and results saved in the system.

(Fig. 4) is an Advanced Security Features Overview—thissectiondescribestheinternalcapabilitiesofScannerand theAdvancedSecurityFeaturesithasaspartoftheScanner platform. It highlights the Scanner's use of machine learning-based detection methods, effective behaviour analysis to evaluate endpoint behaviour, active payload testing,mappingtoOWASPTop10vulnerabilities,andnear real-time scanning capability.

To further describe theoverallworkflowof the Scanner, (Fig. 4). Shows the four stages of the scanning process: 1. reconnaissance and crawling, 2. active payload testing, 3. Innovativemachinelearning-basedanomalyanalysis,and4. report generation.

The live testing of the Scanner resulted in consistent executionofscansandadetailedscanningprogressindicator display as illustrated in (Fig. 5). The average time to scan a medium-sized web application was between 15 to 25 seconds,dependingoncomplexityoftheapplicationandthe number of endpoints discovered. Through the process of quantitative evaluation, it was determined that WA-VScan provided a good detection performance across many differentclassesofvulnerabilities,includinga95%accuracy rate for SQL Injection, 90% for XSS, 85% for Command Injection, and 100% detection rate for Security Misconfiguration vulnerabilities.

The results of the scans are shown in a report generated in the scan results dashboard (Fig. 6), where vulnerabilities areclassifiedaccordingtotheirseveritylevel(Critical,High, Medium, and Low), as well as by OWASP Top 10 (2021) category, and provided with associated CVE (Common Vulnerable Exposures) and CVSS (Common Vulnerability Scoring System) references when available.

WA-VScan performed better than the baseline tools in terms of contextual interpretation of vulnerabilities, and the number of false positives produced by WA-VScan was significantly reduced. This is due to WA-VScan's recruitment of an ensemble anomaly detection technique consisting of an Isolation Forest and One-Class SVM, as well as the anomaly scoring methods based on features in a ranked order, which facilitate the understanding of the vulnerability and increase the level of confidence for the decisions being made about that vulnerability (Fig. 6).

In (Fig. 7), WA-VScan also provides the user with recommendedactionstomitigatedetectedvulnerabilities.In additiontothevulnerabilityentriesinthescanresultsreport, users can find information about how to fix the vulnerabilities, as well as guidance for best practices regardinginformationsecurity,byclickingonthemitigation recommendationlinkslocatedontherightsideofeachentry. All of these entries can be exported to structured PDF files that provide full documentation of the scan, as well as any auditing requirements and compliance-related issues.

To support long-term security monitoring and record keepingofpreviousscansofagivensystem,thescanhistory moduleofWA-VScan keepstrackofallpreviousscansofa given system, and records the summary of how many total scanshavebeenperformed,howmanyofeachseveritylevel, and how the risk has changed over time (Fig. 8).

Finally, the WA-VScan Sample Scan results are illustrated in (Fig. 9 to 13), which show the compiled reporting of all vulnerabilities, classification of the severity of the vulnerabilities, mapping to the OWASP Top 10, and informationonactionableremediationaboutvulnerabilities. These reports provide proof that the system uses raw detection results and transforms that information into a security intelligence that can be utilized on a real-world deployment basis.

In conclusion, the results of the experimental evaluation demonstrate that a hybrid framework that incorporates the use of AI-driven anomaly detection and dynamic payload testing will yield an effective method of identifying, explaining, and ranking the vulnerabilities of web applications in a near real-time manner.

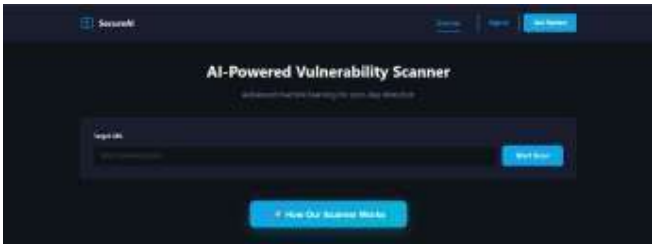


Fig.2.Scanningdashboardshowingreal-timescanprogress

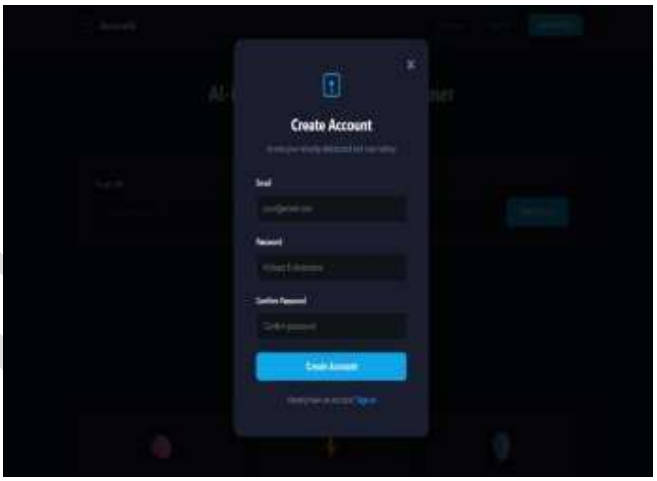


Fig.3.Userregistrationandauthenticationmodule

Fig.4.Livevulnerabilityscanningexecutioninterface

Fig.5.AdvancedsecurityfeaturesoverviewofWA-VScan

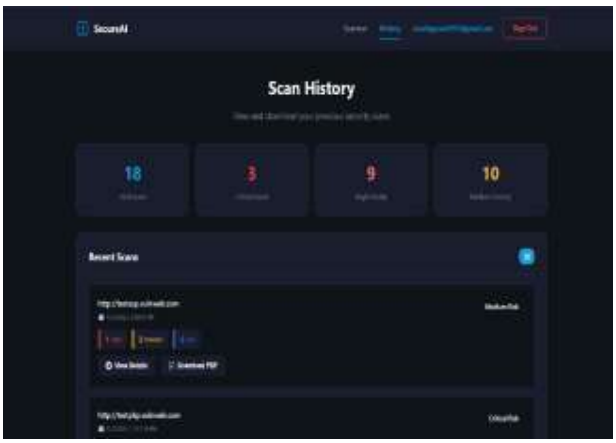


Fig.8.Scanhistorytrackingandrisktrendanalysisdashboard

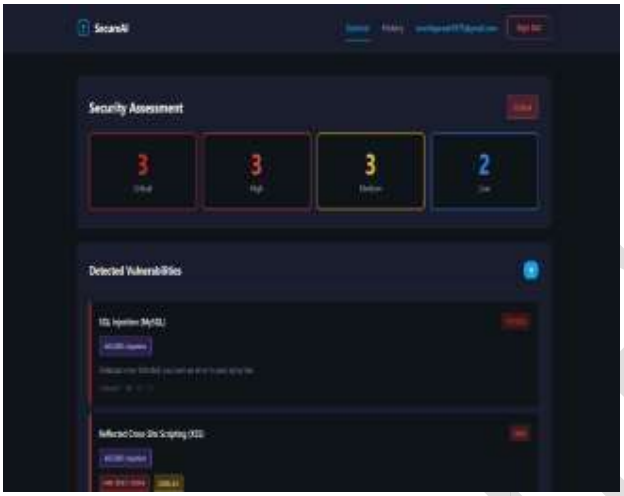


Fig.6.Vulnerabilitydetectionresultswithseverityclassification

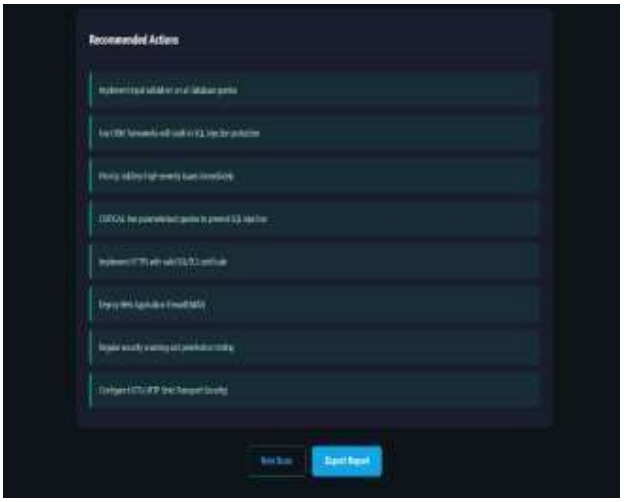


Fig.7.Automatedremediationandmitigationrecommendations

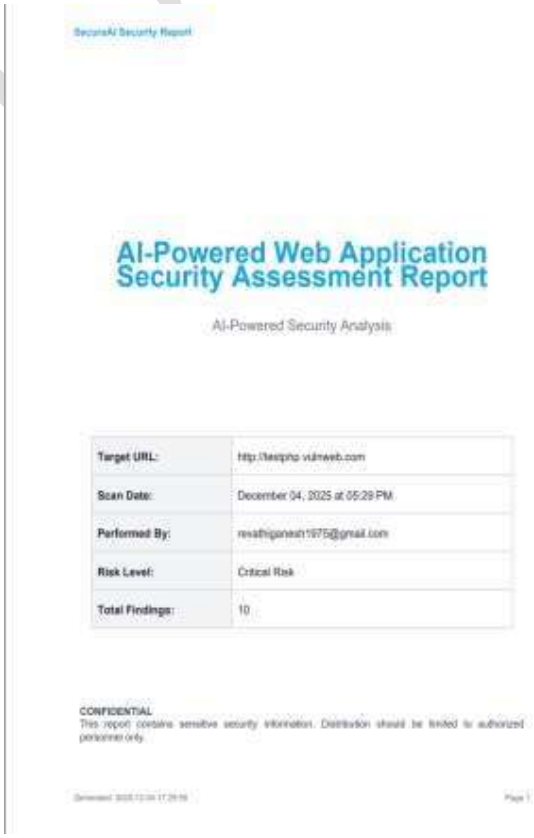


Fig.9.Samplevulnerabilityreportoverview


SecureAI Security Report

EXECUTIVE SUMMARY

RISK OVERVIEW

Severity Level	Count	Status
CRITICAL	2	Immediate Action Required
HIGH	3	Urgent - Address Within 24-48 Hours
MEDIUM	4	Important - Address Within 1 Week
LOW	1	Monitor - Address as Resources Allow

DETAILED FINDINGS

#1	SQL Injection (MySQL)	Critical
Evidence:	Database error detected: you have an error in your sql syntax	
OWASP Category:	A03:2021- Injection	
Target:	http://testphp.vulnweb.com/search.php?test=query	
Test Payload:	' OR '1'='1	

Fig.10.OWASPTop10vulnerabilitymappingresults


SecureAI Security Report

#2	SQL Injection (MySQL)	Critical
Evidence:	Database error detected: you have an error in your sql syntax	
OWASP Category:	A03:2021- Injection	
Target:	http://testphp.vulnweb.com/listproducts.php	
Test Payload:	'	

#3	Reflected Cross-Site Scripting (XSS)	High
Evidence:	Payload reflected unencoded: <script>alert(1)</script>	
OWASP Category:	A03:2021- Injection	
Related CVEs:	CVE-2023-33246 (CVSS: 6.1)	
Target:	http://testphp.vulnweb.com/guestbook.php	
Test Payload:	<script>alert(1)</script>	

#4	Reflected Cross-Site Scripting (XSS)	High
Evidence:	Payload reflected unencoded: <script>alert(1)</script>	
OWASP Category:	A03:2021- Injection	
Related CVEs:	CVE-2023-33246 (CVSS: 6.1)	
Target:	http://testphp.vulnweb.com/guestbook.php	
Test Payload:	<script>alert(1)</script>	

#5	Insecure HTTP Connection	High
Evidence:	Site accessible over unencrypted HTTP	
Target:	http://testphp.vulnweb.com	

Fig.11.CVEandCVSSreferenceintegrationinscanreport


SecureAI Security Report

#6	Missing X-Frame-Options Header	Medium
Evidence:	Clickjacking protection not implemented	
Target:	http://testphp.vulnweb.com	

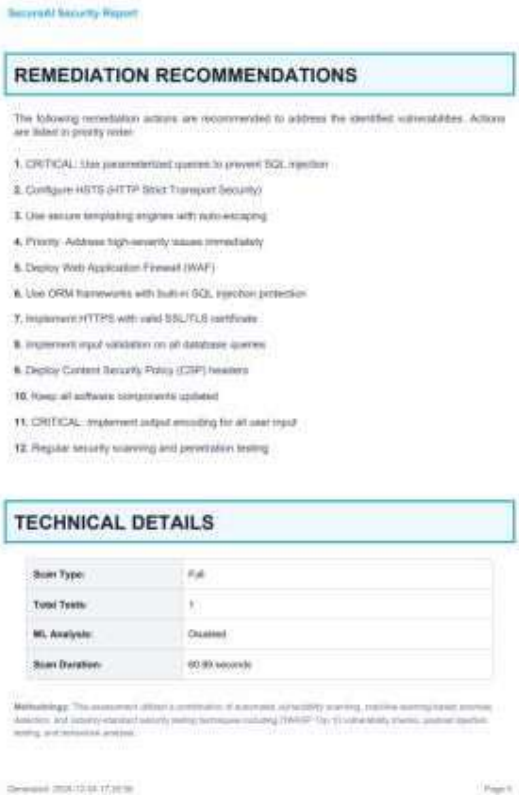
#7	Missing Content Security Policy	Medium
Evidence:	No CSP header found - XSS protection not optimal	
Target:	http://testphp.vulnweb.com	

#8	Potential CSRF Vulnerability	Medium
Evidence:	Form found without CSRF token: search.php?test=query	
Target:	http://testphp.vulnweb.com	

#9	Missing X-Content-Type-Options	Low
Evidence:	MIME-sniffing protection not enabled	
Target:	http://testphp.vulnweb.com	

#10	Server Information Disclosure	Low
Evidence:	Server header reveals: nginx/1.19.0	
Target:	http://testphp.vulnweb.com	

Fig.12.Detailedvulnerabilitybreakdownwithseverityscores


SecureAI Security Report

REMEDIATION RECOMMENDATIONS

The following remediation actions are recommended to address the identified vulnerabilities. Actions are listed in priority order:

1. CRITICAL: Use parameterized queries to prevent SQL injection
2. Configure HSTS (HTTP Strict Transport Security)
3. Use secure templating engines with auto-escaping
4. Priority: Address high-severity issues immediately
5. Deploy Web Application Firewall (WAF)
6. Use ORM frameworks with built-in SQL injection protection
7. Implement HTTPS with valid SSL/TLS certificate
8. Implement input validation on all database queries
9. Deploy Content Security Policy (CSP) headers
10. Keep all software components updated
11. CRITICAL: Implement output encoding for all user input
12. Regular security scanning and penetration testing

TECHNICAL DETAILS

Scan Type:	Full
Total Tests:	1
ML Analysis:	Disabled
Scan Duration:	60.89 seconds

Methodology: This assessment utilized a combination of automated vulnerability scanning, machine learning-based anomaly detection, and industry-standard security testing techniques including OWASP Top 10 vulnerability checks, payload injection testing, and behavioral analysis.

Fig.13.ExportedstructuredPDFvulnerabilityassessmentreport

VI. CONCLUSION

ThisstudypresentedanAI-poweredvulnerabilityscanner that integrates behavioral analytics, ensemble machine learning, and automated intelligence mapping. The proposed solutionimprovestheaccuracyofvulnerabilitydetectionwith ML models that recognize anomalies in real time. It communicates with repositories to integrate resources from OWASPandCVE,makingitnotsimplyanidentificationtool, but a toolthatprovidesdetailedcontextforthreatreports.The executedarchitecturebasedonFlaskisscalable,modular,and could be extended for enterprise-level security testing. The results of the experiment verified the AI-based and hybrid systemyieldsfasteranalysisandcomparablefalsealarmrates than the traditional tools. In conclusion, thescannerprovides the opportunity to fuse state-of-the-art static vulnerability analysis with dynamic and intelligent web application assurance.

VII. FUTUREWORK

Future improvements aim to enhance both scalability and automation. Planned enhancements consist of utilizing multi-threaded crawling mechanisms to conduct scans faster, using reinforcement learning to help build adaptive payloads, and adding support for testing authenticated, session-based applications.Additionally,deeplearningalgorithms(i.e.,Long Short-Term Memory networks and transformers) may be incorporated to improve the anomaly detection mechanism.

Furthermore,integratingthescannerwithreal-timefeedof CVEs, as well as using vulnerability scoring models such as CVSS, will improve the scanner's context capability. The longertermvisionwouldbetopotentially change WA-VScan to a cloud-deployed SaaS tool to monitor and secure web applications worldwide.

VIII. REFERENCES

[1] Yalçinkaya, Mehmet Ali, and Ecir Uğur Küçüksille. "Artificial Intelligence and Dynamic Analysis-Based Web Application Vulnerability Scanner." *ISeCure* 16.1 (2024).

[2] "AI-Based Devadiga, Preeti, et al. "AI-Based Web VulnerabilityScanner:AComprehensiveReview."*Availableat SSRN 5102292* (2025).

[3] Calzavara, Stefano, et al. "Machine learning for web vulnerability detection: the case of cross-site requestforgery." *IEEE Security & Privacy* 18.3 (2020): 8-16.

[4] Ogundairo, O., and P. Broklyn. "Automated Vulnerability Assessment Using Machine Learning." 2024,

[5] Shimmi,Samiha,HamedOkhravi,andMonaRahimi."AI - Based Software Vulnerability Detection: A Systematic Literature Review." *arXiv preprint arXiv:2506.10280* (2025).

[6] Singh, Ravinder, et al. "Analysis of web application vulnerabilitiesusingdynamicapplicationsecuritytesting."*20 24 IEEE 9th International Conference for Convergence in Technology (I2CT)*. IEEE, 2024.

[7] Jana, Indranil, and Alina Oprea. "AppMine: Behavioral analyticsforwebapplicationvulnerability detection."*Proceedingsofthe2019ACMSIGSACConference on Cloud Computing Security Workshop*. 2019.

[8] Pellegrino, Giancarlo, et al. "Deemon: Detecting CSRF with dynamic analysis and property graphs." *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017.

[9] Shahid, Mahnoor. "Machine learning for detection and mitigation of web vulnerabilities and web attacks." *arXiv preprint arXiv:2304.14451* (2023).

[10] M. Amouei, Mohammadhossein, Mohsen Rezvani, and Mansoor Fateh. "RAT: Reinforcement-learning-driven and adaptive testing for vulnerability discovery inweb application firewalls."*IEEE Transactions on Dependable and Secure Computing* 19.5 (2021): 3371-3386.

[11] Zukran, Busra, and Maheyzah Md Siraj. "Performance comparisononsqlinjectionandxssdetectionusingopensource vulnerabilityscanners."*2021InternationalConferenceonData Science and Its Applications (ICoDSA)*. IEEE, 2021.

[12] Fonseca, Jose, Marco Vieira, and Henrique Madeira. "Testing and comparing web vulnerability scanning tools for SQL injection and XSS attacks." *13th Pacific Rim international symposium on dependable computing (PRDC 2007)*. IEEE, 2007.

[13] Hu,Lilan,etal."Webapplicationvulnerabilitydetection method based on machine learning." *Journal of Physics: Conference Series*. Vol. 1827. No. 1. IOP Publishing, 2021.

[14] Singh, Avinash Kumar, and Sangita Roy. "A network based vulnerability scanner for detecting SQLI attacks in web applications."*2012 1st international conference on recent advances in information technology (RAIT)*. IEEE, 2012.

[15] Sonewar,PiyushA.,andSonaliD.Thosar."Detectionof SQL injection and XSS attacks in three tier webapplications."*2016 International Conference on Computing Communication Control and automation (ICCUBEA)*. IEEE, 2016.

[16] Ambedkar, Mohit Dayal, Nanhay Singh Ambedkar, and RamShringarRaw."Acomprehensiveinspectionofcrosssite scriptingattack."*2016internationalconferenceoncomputing, communication and automation (ICCCA)*. IEEE, 2016.

[17] Aliero, Muhammad Saidu, et al. "Classification of Sql InjectionDetectionAndPreventionMeasure."*IOSRJournalof Engineering* 6.02 (2016).

[18] Shukla, Ankur, Basel Katt, and Livinus Obiora Nweke. "Vulnerability discovery modelling with vulnerability severity."*2019 IEEE Conference on Information and Communication Technology*. IEEE, 2019.

[19] Makino, Yuma, and Vitaly Klyuev. "Evaluation of web vulnerability scanners." *2015 IEEE 8th international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS)*. Vol. 1. IEEE, 2015.

[20] Koswara,KevinJonathan,andYudistiraDwiWardhana Asnar. "Improving vulnerability scanner performance in detecting ajax application vulnerabilities."*2019 International Conference on Data and Software Engineering (ICoDSE)*. IEEE, 2019.

[21] Maraj,Arianit,etal."Testingtechniquesandanalysisof SQLinjectionattacks."*20172ndInternationalConferenceon Knowledge Engineering and Applications (ICKEA)*. IEEE, 2017.

[22] AlshuaibiAseel,MohammedAlmaayah,andAitizazAli. "Machine learning for cybersecurity issues: A systematic review." *Secur Challenges* 1.2 (2025).

[23] Y. Li,W. Guo, and J. Zhao, “An Intelligent Framework for Detecting SQL Injection and XSS Attacks Using Deep Learning,”*IEEEAccess*,vol.9,pp.12534–12546,2021.