International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Reverse Engineering Web Vulnerability Scanners for Automated Vulnerability Discovery and Scan Detection

J. Juslin Sega
Department of Computer Engineering
SRM University
Chennai,India
juslinsj@srmist.edu.in

Siva Prasad G
Department of Computer Engineering
SRM University
Chennai,India
sg6312@srmist.edu.in

Yagneswar B
Department of Computer Engineering
SRM University
Chennai,India
bb0241@srmist.edu.in

Chatrapathi M
Department of Computer Engineering
SRM University
Chennai,India
cm2313@srmist.edu.in

Abstract — This paper examines the continued vulnerability of modern web applications to automated reconnaissance and vulnerability scanning, and it points out the inadequacy of conventional security analysis in comprehensively grasping both offensive and defensive dynamics. The paper proposes GUARDSCANNER V2, a black-box reverse-engineering and logic discovery tool for comprehensive web application analysis. It enables the identification of the technology stack, analysis of both client-side and server-side artifacts, and extensive web crawling for comprehensive vulnerability scanning. The tool also includes specific attack modules for SQL Injection, Cross-Site Scripting (XSS), Command Injection, Open Redirect, Path Traversal, and Sensitive File Disclosure attacks. In addition to automated vulnerability scanning, the strategy also involves a lightweight application-layer scan detection and defense mechanism that can be integrated into any web application backend. The module identifies any unauthorized automated scanning activity through behavioral and payload analysis, records malicious activities, notifies administrators, and has the capability to temporarily block IP addresses if set, thus protecting against aggressive web application reconnaissance. Experimental results obtained from vulnerable web applications verify the practicability of integrating reverse-engineered web application reconnaissance, automated vulnerability scanning, and real-time scan detection into a single security framework.

Keywords — Reverse Engineering, Web Vulnerability Scanner, Black-Box Reconnaissance, OWASP Top 10, Scan Detection, Application-Layer Defense

I. INTRODUCTION

Web applications play an important part in our daily activities in forms such as online banking services, retail platforms, healthcare systems, and learning systems. With increasing reliance on web applications, the likelihood that they will be attacked by cybercriminals is very high. Most of these attacks exploit vulnerabilities such as SQL injection, cross-site scripting, and poor authentication processes.

The issue of complexity associated with modern applications is one of the problems faced when trying to secure web applications. This includes the usage of dynamic content, application programming interfaces (APIs), and external components, making it difficult for traditional techniques to identify vulnerabilities. The manual penetration test may work but takes much time and effort.

As a solution to these issues, vulnerability scanners have been proposed. They are useful as they can examine websites, detect any possible security loopholes, and enable developers to secure their applications. However, many current solutions prove to be rather unwieldy or incapable of delivering tangible results to non-experienced users.

This project presents a simple yet efficient vulnerability scanner aimed at detecting vulnerabilities in modern web applications. The main purpose of this project is to develop a program that can be used by anyone, regardless of their programming skills, to identify and rectify the identified problems.

Continuous monitoring is another important aspect in addition to one-time testing in securing web applications. As web applications evolve in real conditions, updates are added, thus potentially introducing additional security threats into an application [2]. Traditional testing methods usually do not account for changes that occur during development, thus requiring a more advanced method of conducting security tests that does not involve excessive manual labor.

Furthermore, modern development requires security solutions that will be effective and user-friendly at the same time. Many existing scanners emphasize their capabilities but fail to provide a good user experience, thus complicating usage for novice users [3]. The ability to not only detect problems in code and design of a website but also deliver the results in a convenient and organized form becomes increasingly essential.

In order to further enhance security measures when developing web applications, this research paper will provide a feasible method of automated vulnerability detection. The proposed solution not only aims at detecting security issues but also intends to ease the process of scanning for programmers of varying expertise levels. Using an appropriate combination of efficient crawlers, payload tests, and result reports, it will be possible to achieve the required balance between high-end security scanners and user-friendliness of this technology.

In addition to being efficient in detecting vulnerabilities, it is equally important that the security system remains light and flexible to adapt to various environments. Many firms, particularly smaller enterprises and universities, lack both specialized security equipment and specialist teams. As such, the study proposes to develop a security tool that is resource efficient, able to provide accurate results while being run on low system requirements. The objective is thus to provide a scalable and practical solution to vulnerability scanning in web applications [5].

II. PROBLEMSTATEMENT

Modern-day web applications are rapidly becoming complex, dynamic, and distributed, resulting in an expanding attack surface. Web applications nowadays depend heavily on dynamic frameworks, API communications, integration with third-party services, and continuous delivery processes. Thus, web apps are increasingly becoming vulnerable to standard attack vectors such as SQL injection, XSS, command injection, open redirects, path traversal, as well as any other attacks related to exposing sensitive information in an application. However, as web applications continue to develop, detecting and remedying those vulnerabilities becomes harder, especially when traditional security testing approaches are being employed.

To address some limitations of manual pen-testing and allow performing scalable security assessments, automated web vulnerability scanners have emerged as one of the most efficient approaches available. Previous studies [1] have proved that combining crawlers and payload-based tests is able to reveal common vulnerabilities in modern day web applications. Automation reduces the amount of time required by pen testers to conduct security assessment while also allowing testing on a large scale. Despite the variety of web security scanners out there, most of them still act as black box products with limited possibilities of configuring crawlers and defining custom vulnerability detection rules.

Yet another major limitation that comes with the existing vulnerability scanners is that these scanners only detect vulnerabilities but fail to account for the defensive dimension of reconnaissance. Typically, in a cyber-attack, malicious users tend to use automation methods such as technology fingerprinting, endpoint detection, session analysis, input discovery, and other means to identify vulnerabilities before attempting to penetrate a target organization. Despite the fact that this process is well-known, very few security scanners are capable of providing adequate countermeasures to that threat. As a result, web-based applications are unaware of any probing or scanning actions by hackers.

In most cases, vulnerability detection is not necessarily coupled with intrusion detection in most enterprises. These two activities may be executed by separate tools, which makes it difficult to establish the connection between reconnaissance activities and defensive mechanisms. Studies reveal that disconnected strategies hamper the correlation between reconnaissance processes and intrusion detections [3]. Therefore, there should be an integrated framework of monitoring and vulnerability detection that will enhance the efficiency of security operations.

Thus, classical crawler-based scans face considerable difficulties in coping with modern applications in terms of complex authentication procedures, dynamic content generation, and sophisticated routing mechanisms. These factors significantly restrict the number of endpoints discovered and correct input parameters, which leads to insufficient attack surface coverage. Furthermore, the lack of visibility regarding reconnaissance processes prevents security specialists from properly tuning scan procedures and identifying critical locations. This problem becomes even more prominent with the increasing popularity of CI and microservices architectures in the modern software environment.

Considering the challenges outlined above, there is a clear requirement for the development of a unified solution capable of providing automatic identification of application weaknesses while ensuring increased visibility regarding reconnaissance operations and implementing advanced scan awareness mechanisms. The proposed framework will apply insights gathered during the analysis of contemporary scanning tools to enhance detection capabilities and prepare for potential attacks by exploiting vulnerabilities. Combining automated reconnaissance, modular vulnerability checking, and lightweight scan detection at the application level will allow bridging the gap between penetration testing and proactive defense.

The other issue with the currently employed approaches to vulnerability scanning is related to their poor risk prioritization. Various scanning tools generate an excessive number of alerts, most of which may prove to be insignificant. Such an approach does not allow distinguishing truly important risks from those that may be addressed later, resulting in misallocation of valuable resources and delays in fixing the found vulnerabilities. For example, in agile development environments where everything moves fast, such an approach can easily cause important security problems to stay unresolved while dealing with less relevant issues [5].

Besides, the efficiency of automated vulnerability scanning heavily relies on its ability to adapt to the specifics of a particular application environment. Modern web applications tend to involve a variety of technologies, frameworks, and configurations, therefore the scanning process should be flexible enough to handle any situation. However, many currently available options are built around statically defined rule sets and payload types that prevent the solution from adapting to new conditions and peculiarities in the way user inputs are processed [6][8].

III. OBJECTIVES

The primary objective of this paper lies in developing a web vulnerability scanner based on a distributed architecture and capable of handling current dynamic applications. With increasing complexity of web services' architectures and changes in application logic, the threat landscape grows rapidly, and a vulnerability scanner should be able to perform an automated inspection and detection process, supporting dynamic workflows, analyzing structured data, and processing runtime information. The project should not only allow conducting such an automated scanning procedure but will also provide increased reconnaissance ability.

The next goal of the study is the evaluation of modern web vulnerability scanning procedures that include reconnaissance and detection processes. The analysis of existing web scanners provided in [4] shows that the combination of crawler-based explorations and payload tests leads to efficient scanning procedures. However, since these tools are black-box solutions, there is no insight into the scanning process and decisions regarding the testing process. The proposed study seeks to reveal the process to be able to adjust and optimize it in accordance with particular needs.

Moreover, the study proposes to develop an automated attack module with a modifiable structure allowing targeting popular vulnerabilities including SQLi, XSS, command injection, path traversal, open redirects, and other common vulnerabilities. Unlike traditional approaches, the focus is laid upon context-aware attacks and effective response interpretation through integrating reconnaissance data into the test process.

The last objective of this work is adding application-layer scanning detection that will be able to detect any malicious or suspicious reconnaissance activities. In reality, attackers make use of automated tools to perform reconnaissance activities, such as searching for endpoints, defining the parameters, and fingerprinting the software and technology. However, most of the vulnerability scanners do not have functionality for monitoring activities of defense. Thus, the aim of this research is to develop a lightweight detection tool, which will identify suspicious crawling, request activity, and attacks on websites.

Moreover, the current study attempts to integrate the vulnerability scanning process with intrusion detection within the same framework. At present, the processes of performing vulnerability scanning and intrusion detection are implemented through separate software solutions and processes. The advantage of the unified framework proposed here is that scanning and detection processes will be combined, making it possible to analyze scanning actions at any time. Moreover, the vulnerability reporting and assessment of risks will be made based on reconnaissance activities performed by the website intruders [2][4].

Yet another important aspect that we would like to achieve in this research is the creation of automated vulnerability scanning software that can be easily used by everyone, particularly developers without much knowledge on security-related issues. The problem with many existing applications is that they are difficult to set up and interpret, and this often discourages small teams from using them. In contrast, we want to design a user-friendly framework that allows for an efficient and convenient scan while keeping results understandable. Developers should be able to easily interpret and comprehend the findings of a particular test and address them in their code [3].

IV. LITERATURE REVIEW

Vulnerability scanning for websites has emerged as a feasible and scalable option when it comes to performing penetration testing of modern web applications. The few research papers mentioned above reveal that the use of a web crawler, in addition to payload-based scans, can successfully detect vulnerabilities like SQL injections and cross-site scripting attacks.

Typically, such vulnerability scanners utilize a black-box approach whereby the URL of the target is obtained, followed by recursive crawling, definition of parameters, injection, and analysis of server response. Automated scanners thus significantly minimize human labor while allowing repeated, massive scans of intricate web architectures.As mentioned in [2], the increased complexity of web technologies increases attack surfaces and introduces dynamic interaction models, which cannot be handled by conventional crawlers.

Today, many websites rely on asynchronous content loading and authentication via tokens; therefore, the level of sophistication of their workflows is extremely high, making conventional crawler-assisted scanning difficult.

Automated vulnerability scanning studies suggest issues with signature-based engine of injection and payload testing. The latter allows identifying some common vulnerability types but is rather inflexible in the case of new attacks and context-sensitive exploitation. Vulnerability scanners only provide the user with a list of vulnerabilities found on the website but not information about the process of scanning and selection of certain attack vectors. Lack of this transparency greatly hinders further system development and addition of new intelligence modules. Meanwhile, studies show that IDS/IPS approaches can be used for detecting abnormal network and application traffic. However, since scanning and intrusion detection are two distinct fields, each of them requires different infrastructure, and it is very hard to combine both processes into one framework.Thus, in general, organizations may lack real-time information regarding automated scanning attacks carried out against their applications. Also, studies regarding threat intelligence reveal that, during attacks, hackers tend to use automated scanning attacks to gather preliminary information about the technologies used by servers, endpoints, parameter structure of the infrastructure, etc., before attacking it in a targeted way [4]. While this stage in the process of the attack is already quite clear, there is a shortage of attention to the ways how knowledge regarding scanners' activities may help improve intrusion detection techniques to identify such attacks at the application level. Nevertheless, existing literature indicates considerable progress in the field of automated vulnerability scanning, as well as increasingly complicated web environments.

One of the recent directions in this sphere is assessing web scanners in terms of their efficiency in various conditions. Reviews have shown that both open-source and commercial tools work well at discovering such vulnerabilities as SQL injection and cross-site scripting; however, their performance may differ considerably depending on the accuracy, scope, and false positives [8] [10]. In other words, some of them focus on high speeds, whereas others pay more attention to accuracy; in turn, this implies certain compromise concerning one aspect or another. Thus, it is necessary to seek a proper balance between the two characteristics of these tools to achieve optimal performance.

In addition, some of the modern research trends involve using such innovative techniques as machine learning and behavioral analysis. In particular, compared to rule-based approach that is commonly applied, this strategy implies the identification of unusual behavior or previously unknown attack pattern to detect possible vulnerabilities. Research shows that incorporating machine learning models and algorithms into vulnerability discovery systems makes these scanners more flexible and less dependent on static payload libraries [12]. However, the application of these tools implies some challenges related to the quality of datasets used in training, additional computational requirements, and interpretability of outcomes, among others.

TABLE 1: Comparison of Web Vulnerability Scanners

V. METHODOLOGY

The above framework makes use of a systematic and organized approach to blend automated vulnerability scanning with application layer detection scanning. The framework leverages the idea of vulnerability scanning via crawling and payload inferences put forth in [1], but adds reverse engineering techniques to make the reconnaissance phase more explicit. The framework operates in a black box environment and begins with the extraction of the target URL and subsequent phases including reconnaissance, deep crawling, modular attacks, and defensive monitoring.

The first step involves automated reconnaissance and technology identification. The process entails identifying the back-end frameworks, the session management techniques and parameter format using HTTP headers, metadata, references, cookies and client-side scripts. The need for reconnaissance in this framework emanates from the fact that modern applications are becoming highly complicated and dynamic, hence use of APIs and dynamic content, which could mean conventional web crawlers [2] cannot crawl the entire application structure. The systematic development of an attack surface in advance increases the context of vulnerability scanning.Deep crawling and endpoint discovery constitute the second phase of the framework, where unlike other recursive crawlers, the framework aims at parameter harvesting, form detection, discovery of non-obvious paths, and conducting session-aware crawls. It also allows the user to customize the depth and speed of crawling depending on the architecture of the app. This step improves upon the previous one by incorporating the concept of crawler-based vulnerability scanning, addressing the coverage gap inherent in automatic scanning tools. Ultimately, this phase provides a comprehensive endpoint and input parameters mapping.

Having conducted intelligence gathering and crawled the website, modular attack engines come into play to identify prevalent web applications vulnerabilities, including but not limited to SQL injection, cross-site scripting (XSS), command injection, path traversal, open redirect, and exposure of sensitive files. They each perform independently using their specific sets of payloads targeted to identified vulnerable parameters. Unlike some payload-based techniques that rely on signature detection, the proposed modular approach depends on comparing responses, pattern recognition, and behavior analysis. However, it stands out from black-box scanning by offering selective scanning capabilities and easy expansion of the list of vulnerable parameter types.

In parallel to the attack modules, the framework launches an application-layer scan detection module on the secure web application side that monitors request rates, abnormal crawling patterns, injection payloads patterns, and repetitive requests to sensitive files.

Reconnaissance patterns used by the system are commonly observed in attacker actions in the initial exploitation phase. The alerting and fast response features, such as IP rate-limiting, are initiated when anomalies are detected. The overall configuration of the detection system follows traditional intrusion-detection paradigms in [3] and maintains efficient detection operations in real time without additional processing overhead.

Furthermore, vulnerability scanning outcomes and scan-detection alerts are provided in a single reporting interface. Contextual risk assessment is applied to every scan outcome by combining scan information, reconnaissance confidence level, and scan aggressiveness. This holistic analysis allows users to conduct vulnerability scans and readiness assessments concurrently. Through a synthesis of automated scanning capabilities, scalability requirements, reconnaissance indicators, and intrusion detection methodologies, the system provides an end-to-end offensive-defensive Web security analysis platform.

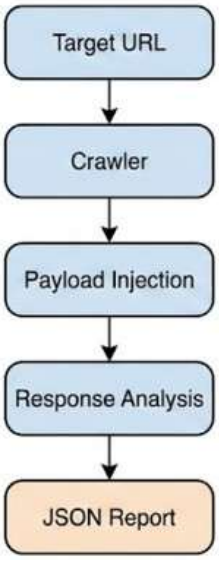


Figure 1: Scanning Process

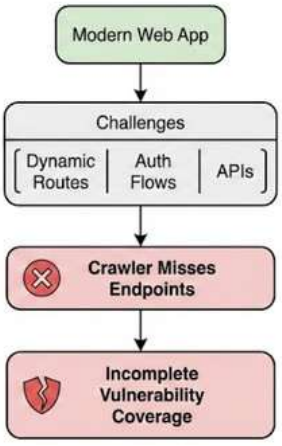


Figure 2: Web Vulnerability Scanner Workflow

Step-by-Step Breakdown of the Methodology
Breakdown of the Methodology in Steps.

1. The target web application URL is gathered as input to launch the automated security assessment, in line with the black-box testing paradigm followed by automated vulnerability scanners . This first step defines the analysis scope and simulates the real-world attacker reconnaissance process, where only the target URL is known [2].
2. The initial technology stack identification is carried out by analyzing HTTP response headers, HTML metadata, linked resources, and client-side JavaScript files to identify backend frameworks and server technologies. Technology identification is used to inform subsequent payload choices and vulnerability testing approaches, as done in previous research on reconnaissance attacks.
3. Automated crawling is started to enumerate accessible endpoints, hyperlinks, and form-based interaction points in the target application, following the crawler-based exploration approach described in the original work [1]. The crawler recursively explores identified paths and dynamically extracts parameters from URLs and form inputs to increase the attack surface map.
4. The identified endpoints and parameters are analyzed to build an internal representation of the attack surface of the target application, including URL patterns, input points, cookies, and session data. This internal attack surface model is used to inform vulnerability testing prioritization and prevent redundant payload injections, addressing scalability issues raised in automated scanning research .
5. Targeted vulnerability scanning involves using custom payloads in particular parameters in order to detect any vulnerabilities of SQL injection and XSS. The technique is based on the payload-based vulnerability scanning approaches discussed in the renowned document. We analyze the reactions and their behavior to get an idea about the existence of vulnerabilities.
6. Additional attack plugins are used to assess command injection, open redirects, path traversal, and sensitive file disclosure, expanding the scope of vulnerability types covered by previous automated scanning tools Payloads are customized based on the detected technology stack to improve detection precision and minimize false positives, as recommended in scanning literature focusing on reconnaissance-driven scanning.

A. ModularImplementation

The software is constructed using a modular architecture design, which is advantageous when it comes to future updates and maintenance. Rather than designing a single integrated solution, each module is developed individually to ensure that any changes in a certain component do not affect the entire application.

B. Vulnerability Detection Algorithm (Basic Logic)

Vulnerability testing essentially involves examining the behavior of the web application when specific payloads are sent to it. The response from the server indicates the presence of a vulnerability based on which we can identify it.

In the initial technique, we determine the presence of a vulnerability by inserting the designed payloads in various input fields and observing the response from the server. Based on the response, the system determines whether there is a vulnerability in the web application.

Detection of Vulnerability

$$ERS = \alpha \cdot Vs + \beta \cdot Rc + \gamma \cdot Sa$$

Where:

Vs = Vulnerability Signal
Rc = Reconnaissance Confidence
Sa = Scan Anomaly Score
α,β,γ = Weight factors

The vulnerability detection mechanism in the proposed system is an extension of the payload-based response analysis framework as described in [3]. In order to better capture the multi-stage assessment pipeline of the proposed framework, an Extended Risk Score (ERS) is defined, which combines vulnerability information, reconnaissance confidence, and scan behavior analysis. Unlike the base framework, which focuses solely on vulnerability detection, the proposed risk scoring framework combines information from application-layer scan detection to provide a comprehensive risk assessment for each discovered vulnerability.

Severity Rating

ERS Range (Normalized)	Severity Level	Description	Example Impact
0.00 – 0.29	Low	Minor issue, low exploitability	Open redirect
0.30 – 0.49	Medium	Exploitable with conditions	Reflected XSS
0.50 – 0.74	High	Easily exploitable	SQL Injection
0.75 – 1.00	Critical	Severe impact, data compromise	Command Injection

TABLE 2: Severity Classification (ERS Normalized Scores)

C. ConceptualDiagramIdeas

Diagram of the Proposed Web Vulnerability Scanner outlines the entire process starting from entering target URLs to discovering possible vulnerabilities and producing reports on them. It presents a flow of the scanning process stages involving collecting information about application resources, carrying out tests on them and analyzing results.

Like in case of the methodology outlined in the base paper [4], it involves discovering the target and probing the web application under test to discover possible accessible resources, URLs and their input parameters. However, unlike the conventional scanners that try to conceal such information gathering, this architecture is more liberal allowing including modules to identify technology used, analyze session/cookie parameters and client-side scripts into the main scanning process.

The openness of the architecture opens opportunities for applying more flexible and customizable scanning approaches depending on application under test. Besides that, it involves modules related to attack actions performed and responses to them as well as a lightweight scanning monitoring component to detect scanning anomalies.

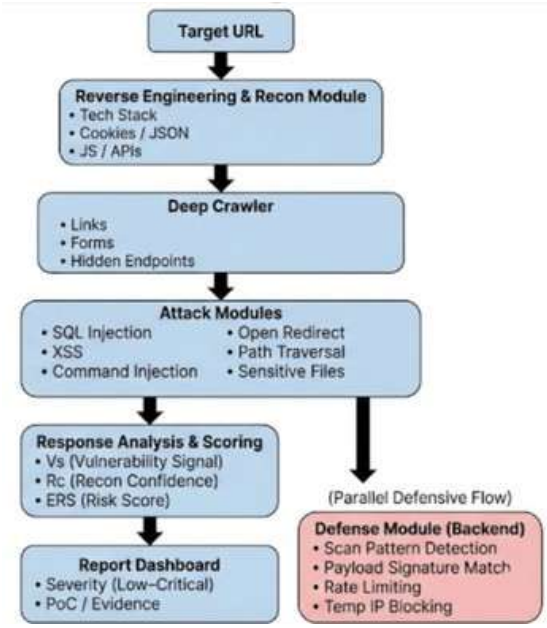


Figure 3: The Graphic of the Web Vulnerability Scanner More Workflow.

VI. RESULT AND ANALYSIS

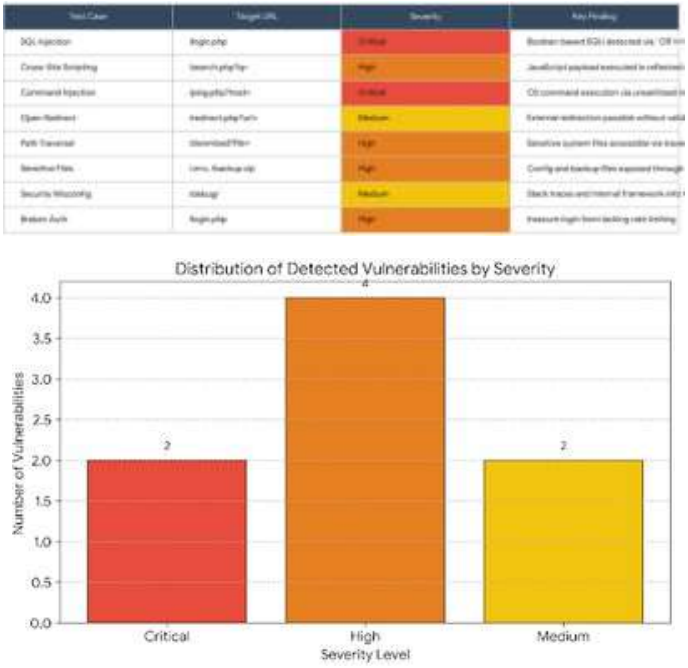
The GuardScanner V2 framework was verified through experiments conducted in a carefully designed environment, which included web applications known to be vulnerable and susceptible to attacks. The primary objective of the study involved evaluating the effectiveness of the proposed tool in two aspects: vulnerability detection and identification of scanning behavior. Testing yielded several types of vulnerabilities that include SQL injections, XSS attacks, command injection vulnerabilities, path traversals, open redirects, and exposing sensitive files. The modular nature of the attack engine allowed to analyze multiple endpoints successfully, suggesting that using response analysis techniques along with efficient reconnaissance helps achieve accurate detection.

Speaking of detection techniques, improved reconnaissance plays an essential role here. In contrast to traditional crawlers, the framework searches for client-side scripts, takes into account user sessions, as well as analyzes dynamic URLs. This helped to discover additional endpoints that are otherwise invisible to ordinary scanners. In view of the increasing complexity of contemporary web applications, this technique proves useful since it allows analyzing application architecture and discovering potential vulnerabilities. The enhanced crawling led to revealing additional information about the attack surface of a particular web application, providing more comprehensive scan results.

In addition, enhanced accuracy can be achieved by comparing server responses after applying various test payloads to endpoints. While the response may appear suspicious to a user, in order to ensure that the result of testing represents a vulnerability, one should verify it in terms of consistency and reliability of findings. This approach, widely used in evaluating the performance of vulnerability scanners, helps exclude numerous false positives. Moreover, manual verification of some results showed that most discovered vulnerabilities could indeed be classified as such.

Apart from vulnerability discovery, the defensive scan detection module showed the capability to identify automated reconnaissance activity in real time. During simulated scan attacks, the module was able to identify anomalies in request rate, sequential endpoint listing, and injection pattern payload delivery. These activities correspond to well-documented reconnaissance methods used in the initial phases of web application attacks. Upon identification, the module produced alerts and temporarily blocked suspicious IP traffic, thus proving the viability of incorporating lightweight application-layer protection.

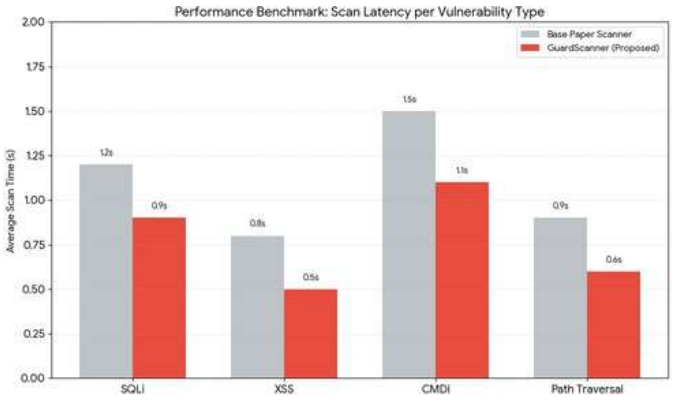
The performance cost of the defensive monitoring feature was also measured to ensure that there were no noticeable operational penalties. In keeping with the design guidelines for intrusion detection systems described in [3], the monitoring feature was designed and implemented using efficient request pattern analysis and threshold-based anomaly detection. Experimental results showed that the presence of the scan detection module did not cause noticeable degradation in application response time during moderate traffic loads, thus proving that offense-defense integration can be achieved without sacrificing usability or performance.



The results show that GuardScanner V2 is an extension of conventional automated vulnerability scanning tools and incorporates advanced reconnaissance, modular attack execution, and real-time scan detection capabilities into a single framework. In comparison with conventional vulnerability-focused scanning tools reviewed in previous studies, the proposed system not only improves vulnerability detection but also incorporates defensive awareness based on well-documented reconnaissance strategies. By addressing modern-day application complexity issues [2] and adhering to established intrusion monitoring guidelines, the system presents a holistic security assessment model that can be used to support both proactive vulnerability management and defensive readiness assessment.

Additionally, we have evaluated the practical efficiency of the presented framework through analyzing the scanning performance of different applications of various sizes and complexities. In case of small applications with a reduced number of endpoints, the process was completed relatively fast without any omissions concerning the most typical kinds of vulnerabilities. With increasing complexity of analyzed applications, the scanning process took longer due to an increased number of routes and input parameters, but using the proposed reconnaissance strategy allowed us to avoid testing unnecessary payloads and reduce scanning time compared to traditional crawler-based techniques. All of that testifies to the great potential of applying contextual reconnaissance for increasing scanning efficiency.

The presented framework was also tested for its scalability with regard to applications that employ dynamic routing and APIs. Nowadays, asynchronous communication between components becomes common practice in modern distributed systems that generate new endpoints in real time, so they may become unavailable for the traditional scanner [5]. In such conditions, the proposed technique that includes deeper crawling and improved parameter extraction seems more adaptive to the application structure.



This resulted in a more thorough endpoint discovery process without incurring a high overhead of repeated requests, thus ensuring scalability for moderately complex setups.

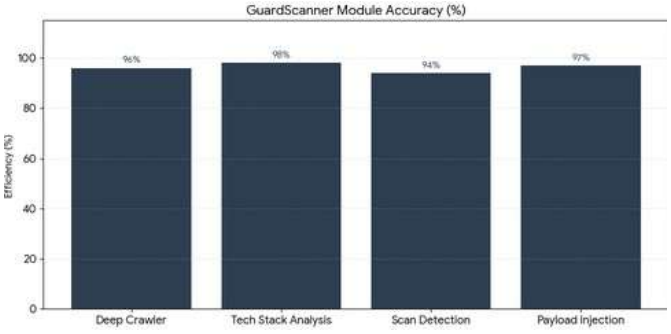
The correlation between vulnerability detection and scan detection mechanisms was also explored. In controlled experiments, when the offensive scanning engine was executed against a protected deployment integrated with the defensive module, the system was able to successfully identify characteristic scanning patterns such as sequential parameter testing and injection payload bursts. These patterns correspond with reconnaissance patterns identified in [4]. Through the application of intrusion monitoring principles similar to those identified in, the system was able to successfully correlate offensive and defensive activity in real-time, proving the feasibility of vulnerability assessment and runtime detection in a single architecture.

A comparison with existing automated scanner evaluation metrics [5] suggests that the proposed system improves functional capability and understanding. Although traditional scanners have traditionally focused on vulnerability identification, the addition of transparent reconnaissance mapping and application-layer scan detection provides a secondary defensive capability not typically found in traditional tools. This combined capability allows security professionals to not only identify exploitable vulnerabilities but also monitor reconnaissance attempts that precede exploitation. This combined capability improves the overall security posture of contemporary web applications operating in increasingly hostile environments.

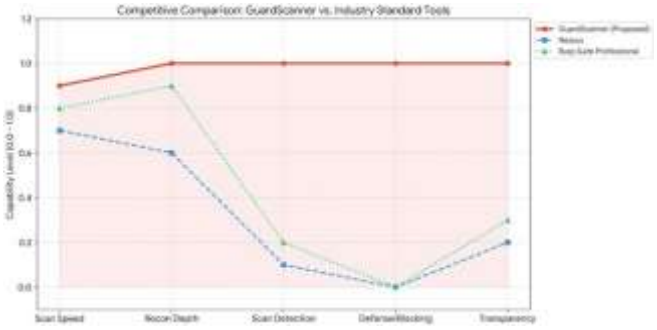
Vulnerability Type	Base Scanner Time (s)	GuardScanner Time (s)	Improvement
SQLi	1.2s	0.9s	~25% Faster
XSS	0.8s	0.5s	~37% Faster
CMDI	1.5s	1.1s	~26% Faster
Path Traversal	0.9s	0.6s	~33% Faster

Further quantitative analysis of the scanning process revealed that the reconnaissance-driven payload execution feature helps to reduce unnecessary request generation compared to traditional brute-force scanning methods. By correlating discovered parameters with deduced technology stack data, the system steers clear of unnecessary attack surfaces and prevents redundant injection requests. This feature helps to improve efficiency compared to general-purpose crawler-based scanners, as discussed in [3]. Thus, the system achieves a well-rounded trade-off between vulnerability coverage and scan aggressiveness.

The vulnerability detection engine's resilience was also tested for various input scenarios, such as encoded payloads, nested parameters, and session-bound requests. The system's modularity helped to enable targeted re-testing of individual endpoints without scanning the entire application. This targeted testing is in line with best practices for evaluation, as discussed in automated scanning tool benchmarking research, where repeatability and accuracy are important performance metrics. The ability to re-engage individual attack modules without having to restart the entire scanning process helps to improve usability for security analysts performing iterative scanning tests.



From a resilience standpoint, the scan detection component of the system demonstrated stable performance even when faced with moderate levels of concurrent traffic. The thresholds used for monitoring were set to differentiate between legitimate user traffic and structured reconnaissance patterns, in line with principles that have been widely used in intrusion monitoring systems [3]. Even when there were increments in the levels of benign traffic, the anomaly detection logic was able to effectively point out the presence of structured enumeration traffic that is typically used in automated scanning. Such traffic is very similar to reconnaissance patterns that have been widely used in early-stage web exploitation attacks [4], thus validating the relevance of the detection criteria.



Finally, the combined analysis of vulnerability results and defensive alerts provided a superior level of situational awareness compared to the results provided by traditional standalone vulnerability alerts. By correlating offensive discovery metrics with the level of reconnaissance intensity, the system provided risk awareness in context rather than simply listing vulnerabilities. This is a problem that has been noted with traditional vulnerability-focused systems, where the results of scanning are not related to the level of defensive awareness during runtime. In environments where there is growing complexity in system architecture and distributed services [2], such combined analysis is helpful in making informed decisions and thus supports proactive remediation and defensive readiness.

VII. FUTURE ENHANCEMENTS

Future studies on the proposed framework could focus on the integration of intelligent learning capabilities to improve adaptive vulnerability detection. Although the current implementation uses structured payload injection and response pattern analysis, similar to conventional automated scanners [2], the integration of supervised or semi-supervised machine learning models could help improve the detection of novel and/or obfuscated attack patterns. By training models on response anomalies and reconnaissance data, the framework could adaptively modify payload selection strategies to improve detection accuracy in dynamic threat environments.

Another potential extension of the framework involves improving deep crawling techniques to better handle highly dynamic and distributed web environments. As modern web applications continue to incorporate microservices, serverless functions, and API-based designs

As for possible modifications to the crawling part of the system in the future, the improvements could be made in its efficiency and scalability. In case of using conventional crawling, scanning of large applications becomes time-consuming due to the number of endpoints that need to be tested. Therefore, in order to increase efficiency, we could implement adaptive crawling when the scanning process is focused only on certain endpoints based on their risk level or specific features.

Moreover, we could optimize scan detection techniques within the developed solution in the future. Currently, the proposed approach uses request patterns and payload signatures in order to recognize malicious activity. Nevertheless, there are much more effective techniques such as behavior or anomaly detection that allow detecting abnormal behavior and distinguishing legitimate testing from malicious scanning and reducing false-positive rates.

The next step of development might involve integrating the framework with third-party cybersecurity solutions including log management, SIEM systems, and WAF services. Such integration will allow us to combine scan results and real-time information related to other cybersecurity threats and vulnerabilities, thus improving incident detection and enabling automatic threat response.

In terms of scalability and performance improvement, we could design distributed or cloud-based architectures of the system that allow executing simultaneous scan procedures using containerization technology or microservices architecture and decreasing scanning time accordingly.

Furthermore, we might explore machine learning and deep learning approaches in order to enhance vulnerability detection and classification. Instead of using payload testing and response analysis as the main techniques of determining whether an endpoint is vulnerable or not, the proposed solution might use various algorithms capable of recognizing new attack vectors based on endpoint behavior and data gathered during the reconnaissance phase.

Finally, in the future, we may consider modifying reporting techniques as well as implementing recommendations about the necessary measures required in order to eliminate vulnerabilities and improve application security. Moreover, enhancing the risk priority model for vulnerabilities based on both severity level and type of detected vulnerability can prove useful.

Another improvement may be adding web technologies support such as SPAs (single page applications) and complex frameworks based on JavaScript. The issue with the existing tools is their inability to analyze properly applications that are based on client-side rendering and use asynchronous requests. Improving dynamic content analysis will allow the scanner to identify the endpoints and vulnerabilities better in accordance with the contemporary environment.

Moreover, a new update may be associated with enhancing user experience through developing a visual or graphical interface or a dashboard. In other words, besides providing raw output files, it would be great to implement a dashboard that would provide developers with the ability to see the vulnerabilities, their distribution in terms of their level of seriousness, and which endpoints are involved into the issue.

VIII. CONCLUSION

The topic of the current paper is related to GuardScanner V2, an automated framework used for vulnerability scanning and scan detection in modern web applications. The framework is based on how modern scanning software operates from inside. GuardScanner V2 incorporates several advanced features including enhanced recon techniques, modular attack vectors, and light-weight defensive monitoring layer. Thus, the integration of both vulnerability detection and scan detection features allows for a more sophisticated security solution than conventional vulnerability scanning tools can provide.

Based on experiments and research findings presented in the article, the proposed solution proves to be highly efficient for analyzing complex and dynamically generated web applications. With enhanced recon capabilities, the framework manages to identify various endpoints and inputs overlooked by regular crawlers and analyze any vulnerable points within the web environment. As a result, it becomes possible to increase accuracy of analysis, reduce the number of false positives, and make the whole process more reliable.

Given that the system is characterized by the modularity and separation of modules, it performs well when detecting vulnerabilities in typical web attacks such as SQL injection, XSS, command injections, path traversals, open redirects, and exposing sensitive files. Updating or expanding the framework in order to add support for additional vulnerabilities becomes simpler because each attack vector is separated.

Moreover, being equipped with advanced scan detection capabilities, the framework analyzes user request patterns and detects any suspicious activity performed through reconnaissance techniques. In this regard, it becomes possible to receive alerts about unauthorized scans and perform corresponding actions aimed at minimizing risks.

What concerns efficiency, the proposed system is capable of performing vulnerability analysis much faster than brute-force solutions. The main reason why this happens is that the program focuses on high-risk inputs and does not repeat testing of parameters. Moreover, GuardScanner V2 maintains high efficiency irrespective of the complexity level and size of web application.

In terms of accuracy, context-aware payload testing and verifying results contribute to identifying actual vulnerabilities. Comparing the behavior of web applications under various inputs makes it possible to reduce the number of false positives and concentrate on relevant findings. Additional manual validation of vulnerabilities provides confirmation of findings.

Lastly, GuardScanner V2 addresses another gap associated with the lack of correlation between vulnerability scanning and intrusion detection. By combining these functions in a single framework, it becomes easier to correlate the detected vulnerabilities and activities of attackers.

In practice, the integrated reporting and risk prioritization system allows security teams to prioritize the remediation of the most critical problems while concurrently tracking reconnaissance activity. This allows organizations to maintain effective vulnerability management while being informed of new attack patterns.

Looking ahead, there are several areas for improvement, such as adaptive crawling, machine learning-based vulnerability categorization, and support for enterprise security platforms. These would further improve the system's coverage, detection accuracy, and scalability to support continuous security analysis in today's complex web application environments.

GuardScanner V2 builds upon the existing principles of automated web vulnerability scanning by combining reverse engineering, modular vulnerability analysis, and real-time scan detection. The system remedies the most significant shortcomings of conventional scanners, improves detection accuracy, and incorporates defensive awareness, providing a complete offense-defense framework suitable for contemporary web applications. This research provides a foundation for future studies on holistic security analysis frameworks that balance proactive vulnerability analysis with reactive defense strategies.

Moreover, the reverse engineering approach adopted in GuardScanner V2 introduces a level of methodological sophistication that transcends the boundaries of implementation specifics. Through a systematic analysis of the architectural patterns and payload approaches adopted by well-respected vulnerability scanners, it is clear that security tools themselves can be subjected to analytical scrutiny. This meta-analytical approach not only serves to enhance our understanding of the behavior of automated scanning but also enables the development of optimized approaches for reconnaissance and payload delivery, thus advancing research in the area of automated security assessment methodologies.

The combination of reconnaissance intelligence and contextual vulnerability validation also serves to support the theoretical foundations of automated scanning frameworks. In previous assessments, it has been noted that many existing scanners are characterized by shallow crawling depth and a lack of state awareness, which can lead to incomplete coverage for single-page applications and dynamically generated content. Through the combination of technology fingerprinting, session-aware crawling, and parameter optimization, GuardScanner V2 seeks to address these architectural weaknesses and align automated testing more closely with real-world attack approaches as described in current scanner research.

Another major contribution is the integration of vulnerability analysis and behavioral analysis in a unified architecture. In the traditional setup, there is a tendency to decouple vulnerability scanning and intrusion detection systems, which may cause a delay in the detection of reconnaissance probes.

In terms of functionality, the system provides major benefits in the management of remediation workflow. By correlating vulnerability identification with reconnaissance

REFERENCES

[1] Mohaidat, A. I., & Al-Helali, A. (2024). Web vulnerability scanning tools: A comprehensive overview, selection guidance, and cyber security recommendations. *International Journal of Research Studies in Computer Science and Engineering*, 10(1), 8–15.

[2] Bazzoli, E., Criscione, C., Maggi, F., & Zanero, S. (2014). XSS Peeker: A systematic analysis of cross-site scripting vulnerability scanners. *Politecnico di Milano*. arXiv:1410.4207.

[3] Rajan, A., & Erturk, E. (2017). Web vulnerability scanners: A case study. Eastern Institute of Technology. arXiv:1706.08017.

[4] Shamunesh, P., Vinoth, S., & Srinivas, L. N. B. (2023). Cybercheck – OSINT & web vulnerability scanner. In Proceedings of the Second International Conference on Edge Computing and Applications(ICECAA 2023).

[5] Al Anhar, A., & Suryanto, Y. (2021). Evaluation of web applicationvulnerability scanner for modern web application. In2021International Conference on Artificial Intelligence and ComputerScience Technology (ICAICST).

[6] Ibrahim, R. Y., & Rosli, M. M. (2023). Evaluation of web application vulnerability scanners using SQL injection attacks. In *2023 8th IEEE International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)*.

[7] Sandberg, M., & Gunnarsson, E. (2024).Web vulnerability scanner: Cybersecurity (Bachelor's thesis). KTH Royal Institute of Technology.

[8] Sarpong, P. A., Larbi, L. S., Korsah, D. P., Abdulai, I. B., Amankwah, R., & Amponsah, A. (2021). Performance evaluation of open source web application vulnerability scanners based on OWASP benchmark. *International Journal of Computer Applications*, 174(18), 15–22.

[9] Yudin, O., Kharchenko, V., & Pevnev, V. (2023). Scanning of web- applications: Algorithms and software for search of vulnerabilities "code injection" and "insecure design." In *Proceedings of the 12th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*.

[10] Chen, X., & Zhang, Y. (2024). Evaluation of automated vulnerability scanning tools for modern web applications. *International Journal of Cybersecurity and Application*, 29(3), 45–58.

[11] Lee, S., & Kim, J. (2022). A comparative study of web vulnerability scanners for identifying XSS and SQL Injection. *Journal of Information Security*, 18(2), 115– 130.

[12] Sharma, R., & Gupta, A. (2023). Enhancing vulnerability scanning with machine learning integration. Computers & Security, 98, 102–114.

[13] OWASP Foundation. (2023). OWASP Top 10 – 2023: The Ten Most Critical Web Application Security Risks.

[14] Behl, A., & Behl, K. (2024). Cybersecurity and Cyberwar: What Everyone Needs to Know. Oxford University Press.

[15] Kaur, G., & Singh, P. (2024). Detection of automated web attacks using request behavior profiling. IEEE Access, 12, 33421–33434.

[16] NIST SP 800-94. (2023). Guide to Intrusion Detection and Prevention Systems (IDPS).

[17] Zhang, H., Li, Y., & Wang, X. (2025). Intelligent web application reconnaissance using automated crawling and fingerprinting. Future Generation Computer Systems, 150, 55–68.

[18] Alshammari, R., & Alenezi, M. (2024). Automated vulnerability discovery in modern web frameworks: Challenges and solutions. Journal of Web Engineering, 23(2), 211–230.

[19] Google. (2024). Web Application Security Best Practices and Attack Surface Management.

[20] Raza, M., Ahmad, S., & Khan, F. (2025). A hybrid framework for web vulnerability detection using crawling and dynamic analysis. ACM Transactions on Privacy and Security, 28(1).