iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.81795

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

A Study on Forensic Reconstruction of Web Application Attacks in a Controlled Xampp Environment

Shwetha. M[1], Dr H R Bhargava[2]
Department of Forensic science, Garden City University, Bangalore, Karnataka, India

Abstract: ***Web applications are an integral part of modern digital systems, supporting activities such as data management, communication, and online services. However, many web applications remain vulnerable to cyber threats due to insecure coding practices, lack of input validation, and weak access control mechanisms. These vulnerabilities can be exploited by attackers to gain unauthorized access, manipulate data, and compromise system integrity. This study focuses on analysing and reconstructing web application attacks using a digital forensic approach in a controlled environment. A vulnerable web application was developed using PHP and MySQL and deployed on a local XAMPP server. The application included basic functionalities such as user authentication and data management, with intentionally introduced security flaws to simulate real-world scenarios. Several controlled attacks, including SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR), were performed to evaluate how these vulnerabilities can be exploited. During the attack execution, digital artefacts were generated in the form of Apache server logs and application logs. These logs contained critical information such as timestamps, request methods, IP addresses, and user inputs. The collected log data was systematically analysed to identify abnormal patterns, including repeated login attempts, suspicious input values, and unauthorized access requests. Based on this analysis, the sequence of attacker actions was reconstructed, providing a clear understanding of how the attacks were carried out step-by-step. The results of this study demonstrate that log analysis plays a crucial role in detecting and investigating web application attacks. Even in compromised systems, logs can serve as reliable forensic evidence to trace attacker behaviour. The study highlights the importance of implementing secure coding practices, proper validation mechanisms, and effective logging systems to enhance web application security and support digital forensic investigations.***

Keywords: Web Application Security, Digital Forensics, SQL Injection, IDOR, Log Analysis, XAMPP

I. INTRODUCTION

Web applications have become an integral part of modern digital systems, enabling services such as online banking, e-commerce, social networking, and data management. Organizations and individuals increasingly rely on web-based platforms for communication, storage, and processing of sensitive information. As the use of web technologies continues to grow, ensuring the security of web applications has become a critical concern (Kumar & Srivastava, 2025; Hu et al., 2020). Despite advancements in development frameworks and security tools, many web applications remain vulnerable due to insecure coding practices, lack of proper input validation, and weak access control mechanisms. These vulnerabilities create opportunities for attackers to exploit systems, gain unauthorized access, and compromise sensitive data. As a result, web application attacks have become one of the most common and significant cyber threats in today's digital environment (Aliero et al., 2020; Al Salmawi, 2025).

One of the most widely exploited vulnerabilities is SQL Injection, where attackers insert malicious SQL queries into input fields such as login forms. If user input is not properly validated, attackers can manipulate database queries to bypass authentication or retrieve confidential information. This allows unauthorized users to gain access to the system without valid credentials (Liu et al., 2020; Zhang et al., 2021; Ahmed et al., 2022). Another common attack is authentication bypass, which occurs when flaws in the login mechanism allow users to access the system without proper verification. This may result from improper handling of user input, weak session management, or insecure authentication logic. Such vulnerabilities can lead to serious security breaches and unauthorized system control (Kritikos et al., 2021; Verma et al., 2023).

In addition, Insecure Direct Object Reference (IDOR) is a critical vulnerability related to access control. In this attack, internal identifiers such as user IDs are exposed in URLs or form parameters. If proper authorization checks are not implemented, attackers can manipulate these identifiers to access or modify other users' data.

This highlights the importance of implementing strong access control mechanisms in web applications (Bakır, 2025; Tadhani et al., 2024). While understanding these attacks is important, it is equally essential to analyse how they can be detected and investigated. Digital forensics plays a crucial role in this process by enabling the collection, preservation, and analysis of digital evidence. In web applications, forensic analysis focuses on examining logs and system-generated data to identify suspicious activities and reconstruct attack events (Shahriar Zawoad & Ragib Hasan, 2020; Chen X. et al., 2023).

Web servers such as Apache generate logs that record all incoming requests and system responses. These logs contain valuable information including IP addresses, timestamps, requested URLs, HTTP methods, and response status codes. By analysing these logs, investigators can identify unusual patterns, track attacker behaviour, and reconstruct the sequence of events during an attack (Kumar S. et al., 2021; Singh A. et al., 2022; Singh K. et al., 2022). For example, repeated login attempts, abnormal input patterns, and unusual URL manipulations in logs may indicate malicious activity. Timestamp analysis further helps in identifying when the attack started, how it progressed, and what actions were performed by the attacker. This information is essential for both forensic investigation and improving system security (Chen X. et al., 2023). In addition to server logs, application-level logs provide deeper insights into user activities such as login attempts, input values, and data modifications. Combining server logs with application logs enhances the accuracy and effectiveness of forensic analysis (Perera D. et al., 2025; Verma et al., 2023).

This study focuses on analysing web application vulnerabilities and reconstructing cyber attacks through a controlled experimental setup. A vulnerable web application was developed using PHP and MySQL within a local XAMPP environment, incorporating basic functionalities such as user authentication and CRUD operations. To simulate real-world scenarios, the application was intentionally designed with security weaknesses. Controlled attacks, including SQL Injection, authentication bypass, and IDOR, were performed to observe their impact and generate forensic artefacts. These artefacts were analysed using log data from the Apache server and the application (Purbawa et al., 2022; Dasari et al., 2025). The analysis enabled the reconstruction of the attack timeline, providing a clear understanding of how the attacks were executed step by step. This demonstrates the practical application of digital forensic techniques in investigating web-based cyber attacks (Yu et al., 2026).

The significance of this study lies in its practical approach to understanding both web application security and digital forensics. It highlights how vulnerabilities can be exploited and how forensic analysis can be used to trace attacker activities. The study also emphasizes the importance of secure coding practices, proper input validation, and strong access control mechanisms. Furthermore, it underlines the importance of maintaining effective logging systems, as logs serve as critical evidence in detecting and investigating cyber incidents. Without proper logging, it becomes difficult to trace attacker behaviour or reconstruct attack events (Perera I. et al., 2025). This study bridges the gap between web attack execution and forensic reconstruction by analysing real-time log evidence in a controlled environment.

II. REVIEW OF LITERATURE

A. *Review of Literature*

The rapid growth of web applications for communication, data storage, and online services has significantly increased exposure to cyber threats. Web applications are frequently targeted due to vulnerabilities arising from insecure coding practices, improper input validation, and weak authentication mechanisms. Among various web-based threats, SQL Injection (SQLi) and Cross-Site Scripting (XSS) are considered the most critical due to their ability to compromise sensitive data and system integrity.

Kumar and Srivastava (2025) provided a comprehensive review of SQL injection attacks, discussing various techniques used by attackers and their impact on web applications. The study emphasized the importance of secure coding practices such as input validation and parameterized queries. It also highlighted future research directions in improving detection mechanisms. The authors concluded that prevention at the development stage is more effective than post-attack mitigation.

Mutedi and Tjahjono (2022) conducted a systematic literature review on preventing SQL injection attacks using OWASP-based web application firewalls. Their study showed that firewalls can effectively filter malicious inputs and block common attack patterns. However, they also noted that these systems require frequent updates to remain effective. The research emphasized the importance of combining firewall protection with secure coding practices.

Bakır (2025) proposed UniEmbed, a machine learning-based framework designed to detect both SQL injection and Cross-Site Scripting (XSS) attacks. The model uses embedding techniques to understand input patterns and identify malicious behaviour. The results demonstrated improved detection accuracy compared to traditional methods. This study highlights the importance of unified detection systems for multiple web vulnerabilities.

Chen, Liang, and Wang (2025) developed a hybrid SQL injection detection model that integrates content matching techniques with deep learning. Their approach improved detection accuracy and reduced false positives. The study demonstrated that combining traditional and modern techniques enhances system robustness. It also showed effectiveness in detecting complex and obfuscated attack patterns.

Al Salmawi (2025) critically evaluated existing SQL injection security measures in web applications. The study identified major limitations in traditional defence techniques, especially against advanced attack methods. It emphasized the need for improved security mechanisms and continuous monitoring. The research concluded that current solutions must evolve to address modern cybersecurity challenges.

Purbawa et al. (2022) proposed an ensemble-based SQL injection detection method that combines multiple machine learning algorithms. The model achieved higher accuracy and reduced false positives compared to single-model approaches. The study demonstrated that ensemble techniques are effective in handling complex attack patterns. It also highlighted the importance of combining different algorithms for better performance.

Kritikos et al. (2021) explored deep learning techniques for SQL injection detection and compared them with traditional approaches. The study found that rule-based systems are limited in detecting unknown attacks. Deep learning models, on the other hand, showed better adaptability and accuracy. The research emphasized the need for intelligent systems in modern cybersecurity.

Liu, Li, and Chen (2020) introduced DeepSQLi, a deep learning-based approach that analyses the semantic structure of SQL queries. The model was able to identify malicious inputs by understanding query context. The results showed high accuracy in detecting SQL injection attacks. This study demonstrated the effectiveness of semantic analysis in cybersecurity.

Dasari et al. (2025) proposed the use of generative models to enhance SQL injection detection and prevention. Their approach can simulate attack patterns and improve system training. The study showed that generative models can increase robustness against unknown attacks. It also highlighted their potential in improving cybersecurity systems.

Yu et al. (2026) developed a multi-agent honeypot-based dataset for SQL injection detection. The dataset captures real-world attack scenarios, making it valuable for training machine learning models. The study emphasized the importance of realistic data in improving detection accuracy. It also contributes to research in attack simulation.

Perera I. et al. (2025) applied machine learning and NLP techniques to detect malicious SQL queries. Their model analysed input patterns and query structures to identify attacks. The study showed improved detection efficiency compared to traditional methods. It also highlighted the role of NLP in cybersecurity applications.

Hu, Zhao, and Cui (2020) presented a survey on SQL injection attacks, including detection and prevention methods. The study discussed various techniques such as input validation, firewalls, and secure coding. It provided a detailed overview of existing solutions and their limitations. The authors emphasized the need for multi-layered security approaches.

Aliero et al. (2020) focused on detecting SQL injection vulnerabilities in web applications. Their research showed that improper input validation is a major cause of security issues. The study highlighted common vulnerabilities in web systems. It emphasized the importance of secure coding practices.

Zhang et al. (2021) applied LSTM-based deep learning models to detect web application attacks. The study showed that sequential learning improves detection accuracy. It was effective in identifying complex and evolving attack patterns. The research demonstrated the importance of deep learning in cybersecurity.

Ahmed et al. (2022) proposed a CNN-based approach for detecting SQL injection attacks. The model was able to identify hidden patterns in malicious inputs. The results showed improved accuracy compared to traditional methods. The study highlighted the effectiveness of CNN in detecting complex attacks.

Verma et al. (2023) developed a real-time detection system for web application attacks using machine learning. The system was capable of detecting multiple attack types and generating alerts. The study demonstrated the effectiveness of real-time monitoring. It also emphasized the importance of automated detection systems.

Singh A. et al. (2022) proposed a log-based intrusion detection system for web applications. The study showed that analysing server logs can help identify suspicious activities. It highlighted the importance of log analysis in cybersecurity. The system was effective in detecting abnormal behaviour.

Kumar S. et al. (2021) analysed Apache server logs to detect cyber attacks. The study identified abnormal request patterns as indicators of malicious activity. It demonstrated how logs can be used for forensic analysis. The research emphasized the importance of logging mechanisms.

Gupta et al. (2021) focused on client-side detection of Cross-Site Scripting (XSS) attacks. The study showed that browser-based mechanisms can prevent script execution. It highlighted the importance of client-side security. The research demonstrated effective XSS prevention techniques.

Singh R. et al. (2022) used NLP techniques to detect XSS attacks. The model analysed input patterns and scripts to identify malicious behaviour. The study showed improved detection accuracy. It highlighted the role of NLP in web security.

Tadhani et al. (2024) proposed a deep learning-based system for detecting web vulnerabilities. The model was capable of identifying multiple attack types. The study showed high accuracy and efficiency. It emphasized the importance of AI in cybersecurity.

Babaey et al. (2025) developed an AI-based framework for web attack detection. The study highlighted the role of artificial intelligence in improving detection systems. It demonstrated better performance compared to traditional methods. The research supports AI-based security solutions.

Zawoad and Hasan (2020) emphasized the importance of digital forensics in web applications. Their study showed how logs and artefacts can reconstruct attacker behaviour. It highlighted the role of forensic analysis in investigations. The research is important for understanding cyber attacks.

Chen X. et al. (2023) focused on forensic investigation using log data. The study demonstrated how logs can help reconstruct attack sequences. It provided insights into attacker behaviour. The research highlighted the importance of forensic artefacts.

Singh K. et al. (2022) explored web log analysis for intrusion detection systems. The study showed that pattern recognition techniques can detect anomalies. It improved detection accuracy. The research emphasized log-based security systems.

Perera D. et al. (2025) proposed an AI-based forensic analysis approach for web logs. The system enabled automated detection of malicious activities. It improved scalability and efficiency. The study highlighted the future of AI in forensic analysis.

B. Research Gap

Although numerous studies have explored detection and prevention mechanisms for web application attacks, limited attention has been given to the forensic reconstruction of such attacks using real-world experimental environments. Most existing works emphasize machine learning-based detection techniques without analysing the artefacts generated during actual attack execution. Digital forensic investigations rely heavily on system logs, server records, and application traces to reconstruct cyber incidents. However, there is a lack of practical studies demonstrating how these artefacts are generated and how they can be systematically analysed to trace attacker behaviour.

Furthermore, existing research rarely integrates multiple attack types such as SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR) within a single framework. This limits the understanding of how different vulnerabilities interact and how attackers exploit them collectively. There is a clear need for experimental research that combines attack simulation, log analysis, and forensic reconstruction in a controlled environment. This study addresses this gap by developing a vulnerable web application, performing controlled attacks, and analysing the resulting forensic artefacts to reconstruct attacker behaviour.

III. OBJECTIVES

A. Aim

The primary aim of this study is to analyse and reconstruct web application attacks through a forensic perspective using a controlled experimental environment. The research focuses on understanding how common web vulnerabilities can be exploited and how the resulting digital artefacts can be used to trace and investigate attacker activities.

The study also aims to demonstrate the importance of logging mechanisms and forensic analysis in identifying security breaches within web applications.

B. Objectives Of The Study

The following objectives are formulated to achieve the aim of this research:

1) To design and develop a vulnerable web application using PHP and MySQL in a controlled XAMPP environment.
2) To simulate common web application attacks such as SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR).
3) To observe and record the digital artefacts generated during the execution of these attacks.
4) To analyse Apache server logs and application logs to identify traces of malicious activities.
5) To reconstruct the sequence of attacker actions based on log analysis and forensic evidence.
6) To examine how vulnerabilities in web applications can be exploited to gain unauthorized access.

7) To evaluate the role of logging mechanisms in detecting and investigating cyber attacks.
8) To highlight the importance of secure coding practices and input validation in preventing web application attacks.

C. Scope of the Study

This study is conducted within a controlled experimental environment using the XAMPP server setup. The research is limited to the development of a basic web application with intentionally introduced vulnerabilities for educational and forensic analysis purposes.
The scope includes simulation of selected web attacks such as SQL Injection, authentication bypass, and IDOR. The study focuses on analysing server-side artefacts, particularly Apache logs and application logs, generated during these attacks.
The research does not involve real-world attack deployment or unauthorized system access. It is strictly limited to controlled testing for academic and research purposes. Additionally, advanced attack techniques and large-scale intrusion detection systems are beyond the scope of this study.

D. Significance of the Study

Web applications are widely used across various domains, including banking, healthcare, education, and e-commerce, making them prime targets for cyber attacks. Understanding how these attacks occur and how they can be traced is essential for improving cybersecurity practices.
This study is significant as it provides practical insights into the forensic reconstruction of web application attacks. By analysing real attack simulations, the research helps in understanding how digital evidence is generated and how it can be used during investigations.
The findings of this study can assist students, researchers, and cybersecurity professionals in gaining a deeper understanding of web application vulnerabilities and forensic analysis techniques. It also emphasizes the importance of implementing secure coding practices and effective logging mechanisms to enhance application security.

IV. METHODOLOGY

A. Materials

The experimental setup for this study required both hardware and software components to develop the web application, simulate attacks, and perform forensic analysis. A personal computer running the Windows operating system was used as the primary platform for conducting the experiment. The web application was developed using PHP as the server-side scripting language and MySQL as the database management system. The XAMPP server environment was used to host the application locally, as it provides an integrated platform consisting of the Apache web server, MySQL database, and PHP interpreter. The application included a basic login system and CRUD (Create, Read, Update, Delete) functionalities to simulate real-world web application behavior. These features were intentionally designed with security vulnerabilities to allow controlled attack simulations.
For forensic analysis, Apache server logs were used to monitor HTTP requests and user activities. Application-level logs were also generated to capture user actions such as login attempts and data modifications. Additionally, tools such as the XAMPP control panel and browser developer tools were used to observe system behavior during attack execution. All experimental observations, including log outputs and screenshots, were documented systematically for further analysis.

B. Methodology

This study follows an experimental approach to analyse the forensic reconstruction of web application attacks within a controlled environment. Initially, a vulnerable web application was designed and developed using PHP and MySQL. The application included essential functionalities such as user authentication and data management operations. Security weaknesses were deliberately introduced in the application, such as lack of input validation and improper access control mechanisms, to simulate real-world vulnerabilities. Once the application was deployed on the XAMPP server, multiple controlled attack simulations were performed. These attacks included SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR). During the execution of these attacks, the system generated various digital artefacts in the form of server logs and application logs. The Apache server logs were analysed to track incoming HTTP requests, including GET and POST methods, timestamps, IP addresses, and requested URLs. These logs provided valuable insights into how the attacker interacted with the application. In addition to server logs, application logs were examined to identify specific user actions such as login attempts, failed authentications, and data modifications. These logs helped in understanding how vulnerabilities were exploited within the application.

The collected log data was then analysed to reconstruct the sequence of events during the attack. By correlating timestamps and request patterns, it was possible to identify suspicious activities and trace the behavior of the attacker.
This methodological approach enabled the identification of forensic artefacts and demonstrated how web application attacks can be reconstructed using log analysis techniques.

C. *Experimental Workflow*

1) Development Phase: A vulnerable web application was developed using PHP and MySQL within the XAMPP environment, including login and CRUD functionalities.
2) Vulnerability Introduction: Security weaknesses such as lack of input validation and improper access control were intentionally introduced.
3) Attack Simulation: Controlled attacks including SQL Injection, authentication bypass, and IDOR were performed.
4) Data Collection: Digital artefacts were collected from Apache server logs and application logs.
5) Log Analysis: The collected logs were analysed to identify suspicious patterns and malicious activities.
6) Forensic Reconstruction: Th sequence of attacker actions was reconstructed based on log evidence .
7) Result Interpretation: The findings were analysed to understand how vulnerabilities were exploited and how forensic artefacts support investigation.

D. *Tools and Technologies*

The following tools and technologies were used:
1) PHP (Server-side scripting)
2) MySQL (Database management system)
3) XAMPP (Apache server environment)
4) Apache Web Server (Log generation)
5) Web Browser (Client interaction)
6) XAMPP Control Panel (Server management)

V. RESULTS AND DISCUSSION

A. *Experimental Environment*

The experiment was conducted in a controlled local server environment using XAMPP on a Windows operating system. The web application was hosted on the Apache server and connected to a MySQL database. The application included functionalities such as user login and CRUD operations, which were intentionally designed with security vulnerabilities.
The system successfully simulated real-world web application behaviour, allowing controlled execution of various cyber attacks and generation of forensic artefacts.

B. *Execution of Web Application*

The developed web application was successfully deployed and accessed through a web browser. Users were able to perform operations such as login, viewing records, editing user data, and updating information.
During normal usage, the application generated standard HTTP requests recorded in the Apache logs. These requests included GET and POST methods along with timestamps and requested URLs.

Figure 5.2.1: Web Application Interface Showing Login and User Operations.

The above figure shows the successful execution of the web application with login and data management functionalities.

C. SQL Injection Attack Analysis

SQL Injection attacks were performed by inserting malicious SQL queries into input fields such as the login form. The attack aimed to bypass authentication and gain unauthorized access to the system. The results showed that the application was vulnerable to SQL Injection due to lack of input validation. Unauthorized login access was successfully achieved.

The Apache logs recorded suspicious requests containing unusual query patterns. For example:

- Repeated POST requests to the login page
- Abnormal input patterns in request parameters
- Multiple login attempts within a short time interval

These log entries served as important forensic artefacts indicating malicious activity.

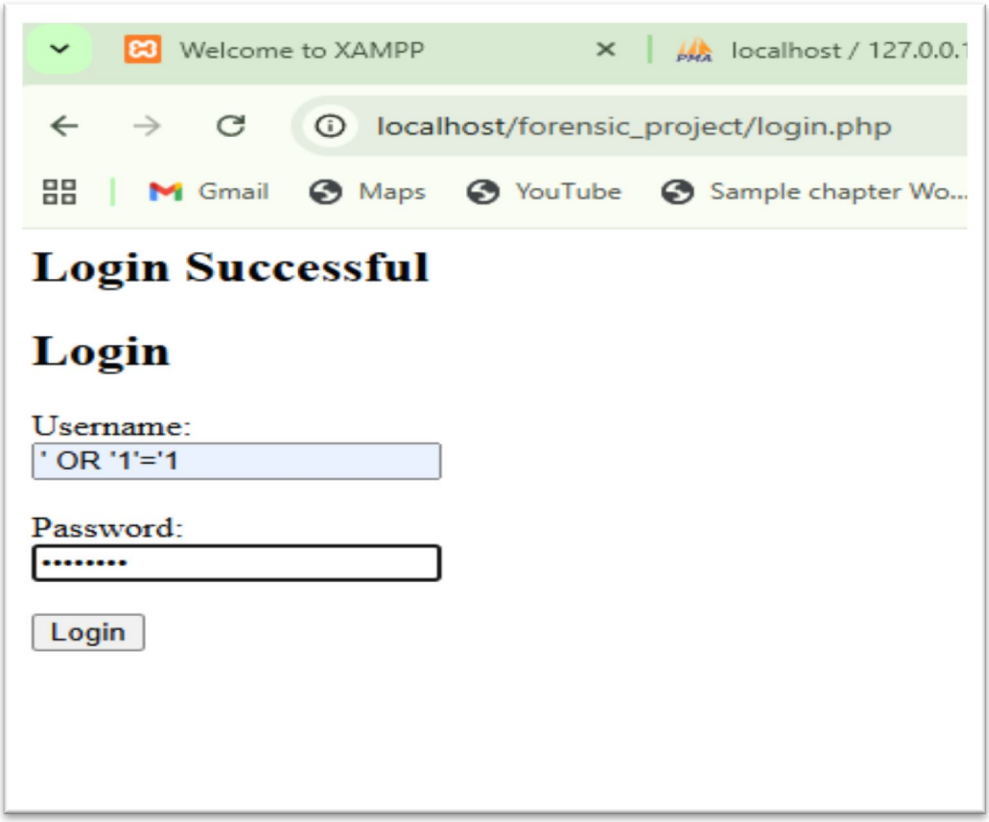


Figure 5.3.1: SQL Injection Attack Performed on Login Field

The above figure illustrates the execution of a SQL Injection attack by entering a malicious query in the login field, resulting in unauthorized access to the application

D. Authentication Bypass Analysis

Authentication bypass attacks were performed to gain access to the application without valid credentials. This was achieved by manipulating input fields or exploiting logical flaws in the authentication mechanism. The results confirmed that weak authentication logic allowed unauthorized access. The system failed to properly verify user credentials, leading to successful login attempts without valid authentication.

Log analysis revealed:

- Multiple failed login attempts followed by successful access
- Repeated POST requests to the login endpoint
- Absence of proper validation checks

These observations highlight the importance of secure authentication mechanisms.

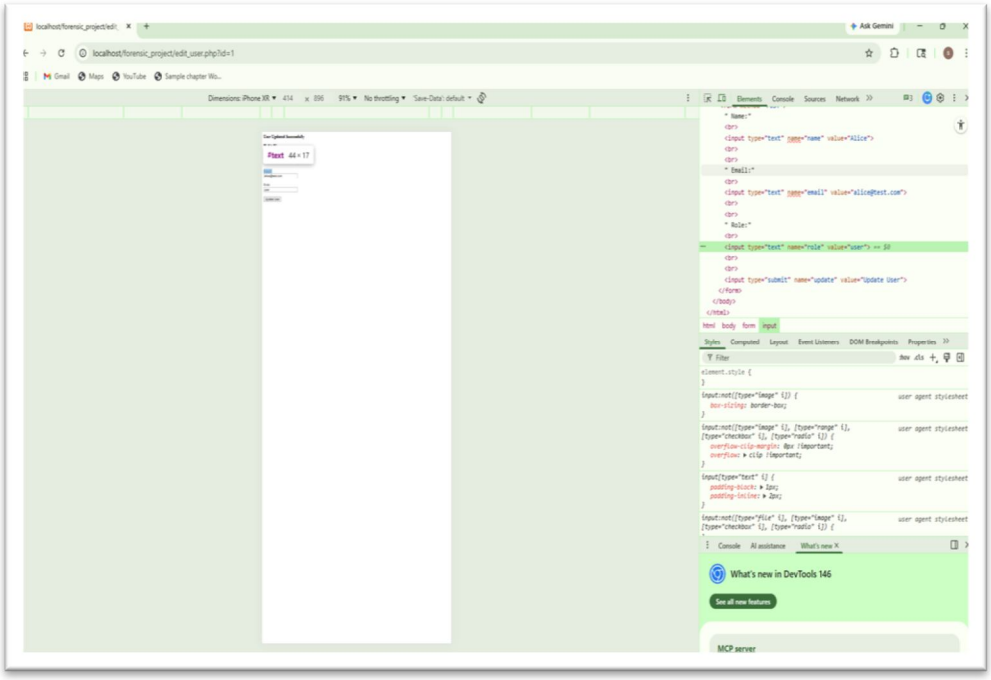


Figure 5.4.1: Successful Authentication Bypass Without Valid Credentials

The above figure demonstrates how the application allows access without proper authentication, indicating a flaw in the login validation mechanism.

E. IDOR (Insecure Direct Object Reference) Analysis

IDOR attacks were performed by modifying URL parameters to access unauthorized user data. By changing user IDs in the URL, it was possible to view or edit data belonging to other users.

The application failed to implement proper access control, allowing unauthorized data access. Apache logs showed:

- GET requests with modified user ID parameters ,Direct access to restricted resources
- Sequential access to multiple user records

These patterns indicate unauthorized data access attempts and serve as key forensic evidence.

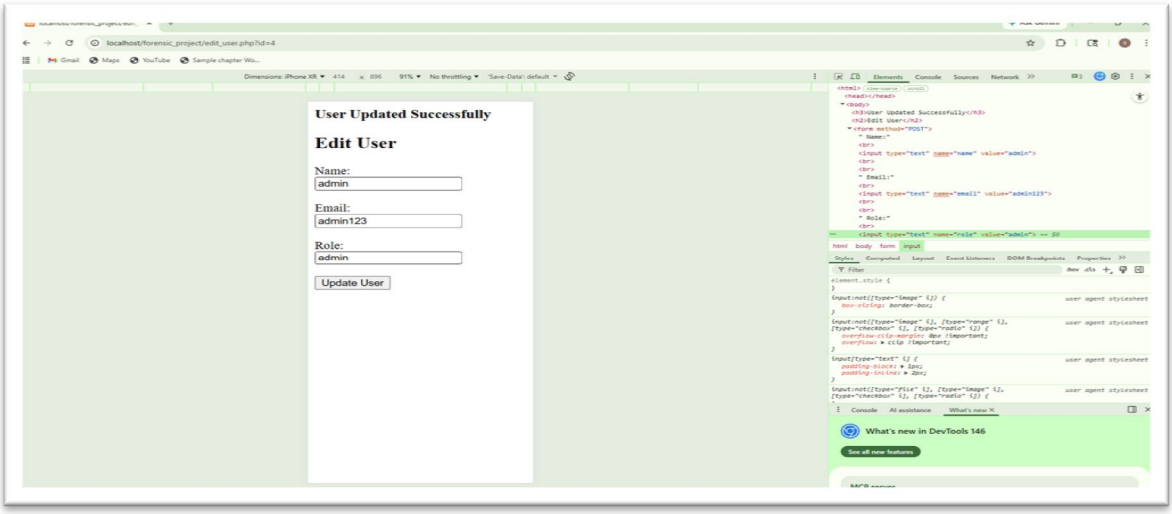


Figure 5.5.1: Accessing User Data with Original User ID=4

The above figure shows the normal access of user data using a valid user ID.

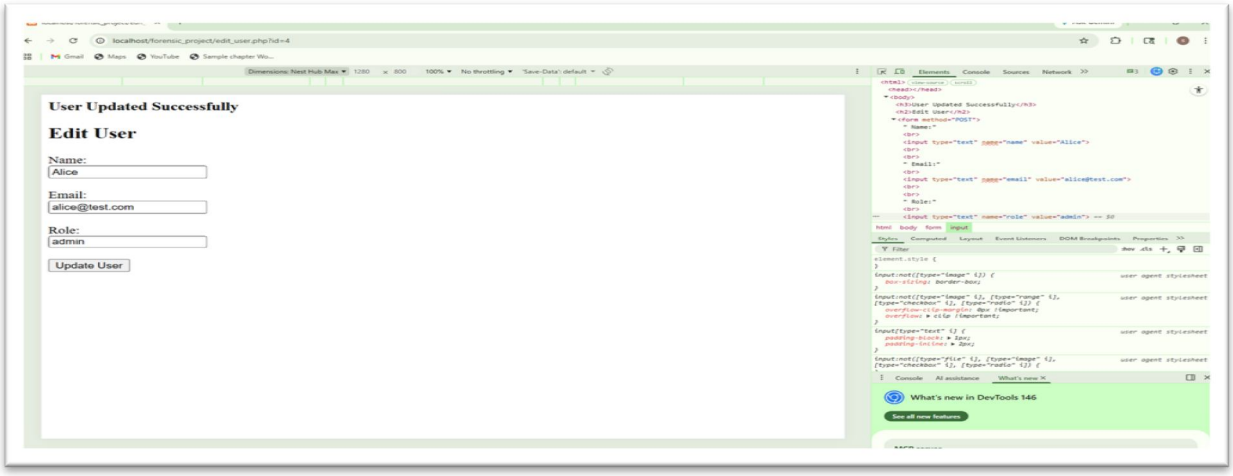


Figure 5.5:2 Unauthorized Access by Modifying User ID Parameter

The figure demonstrates how modifying the user ID in the URL allows access to another user's data without proper authorization, indicating an IDOR vulnerability.

F. Log Analysis and Artefact Identification

The Apache server logs played a crucial role in identifying and analysing attack patterns. Each log entry contained important information such as:

- IP Address (e.g., ::1 for localhost)
- Timestamp of request
- HTTP request method (GET/POST)
- Requested URL
- Response status code

Example log entries observed:

GET/forensic_project/login.php

POST/forensic_project/login.php

GET/forensic_project/edit_user.php?id=4

POST /forensic_project/edit_user.php?id=4

These logs clearly show how the attacker interacted with the application.

From the analysis, the following forensic artefacts were identified: Repeated login attempts, Suspicious POST requests, URL manipulation patterns ,Unauthorized access attempts

```
http://localhost/forensic_project/login.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/145.0.0.0 Safari/537.36"
:1 - - [10/Mar/2026:13:59:15 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
http://localhost/forensic_project/login.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/145.0.0.0 Safari/537.36"
:1 - - [10/Mar/2026:13:59:47 +0530] "POST /phpmyadmin/index.php?route=/ HTTP/1.1" 200
573 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/145.0.0.0 Safari/537.36"
:1 - - [10/Mar/2026:14:06:41 +0530] "POST /phpmyadmin/index.php?route=/ HTTP/1.1" 200
573 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/145.0.0.0 Safari/537.36"
:1 - - [10/Mar/2026:14:21:23 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:21:26 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:21:29 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:21:40 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:21:48 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:22:27 +0530] "POST /phpmyadmin/index.php?route=/ HTTP/1.1" 200
573
:1 - - [10/Mar/2026:14:27:04 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 252
:1 - - [10/Mar/2026:14:27:14 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:27:27 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:27:28 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 252
:1 - - [10/Mar/2026:14:27:38 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:28:56 +0530] "POST /phpmyadmin/index.php?route=/ HTTP/1.1" 200
573
:1 - - [10/Mar/2026:14:32:22 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
:1 - - [10/Mar/2026:14:32:33 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 252
:1 - - [10/Mar/2026:14:32:48 +0530] "POST /forensic_project/login.php HTTP/1.1" 200 256
```

Figure 5.6.1: Apache Server Log Entries Showing Repeated POST Requests

The above figure shows the Apache server log entries generated during the execution of the web application. It can be observed that multiple POST requests were made to the login page (/forensic_project/login.php) within a short time interval. These repeated requests indicate possible malicious activity, such as SQL injection or authentication bypass attempts.

Additionally, the presence of requests to /phpmyadmin/index.php and continuous login attempts suggests abnormal user behaviour. The timestamps and request patterns clearly demonstrate how the attacker interacted with the system. These log entries serve as important forensic artefacts for identifying suspicious activity and reconstructing the sequence of the attack.

G. Attack Reconstruction

By analysing the collected logs, it was possible to reconstruct the sequence of attacker actions. The typical attack flow observed was:

1) Accessing the login page
2) Attempting multiple login requests
3) Injecting malicious input for authentication bypass
4) Gaining unauthorized access
5) Modifying user data through edit functionalities
6) Accessing different user records using IDOR

This reconstruction demonstrates how attackers exploit vulnerabilities step-by-step.

VI. DISCUSSION AND CONCLUSION

A. Discussion

The findings of this study demonstrate that vulnerable web applications developed using PHP and MySQL in a controlled XAMPP environment can be easily exploited when proper security measures are not implemented. The successful execution of attacks such as SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR) highlights the critical weaknesses present in poorly designed web systems.

The simulation of these attacks provided practical insight into how attackers manipulate input fields and URLs to gain unauthorized access. It was observed that the absence of input validation and weak authentication mechanisms allowed attackers to bypass login systems and directly access restricted data. This emphasizes the importance of secure coding practices during the development phase.

A significant aspect of this study was the observation and recording of digital artefacts generated during the attacks. Apache server logs and application logs proved to be valuable sources of forensic evidence. These logs contained crucial information such as IP addresses, request methods, timestamps, and accessed resources, which helped in identifying malicious activities.

Through detailed log analysis, it was possible to reconstruct the sequence of attacker actions. The correlation between different log entries enabled a step-by-step understanding of how the attack was carried out, from initial access attempts to successful exploitation. This demonstrates the importance of maintaining proper logging mechanisms for effective digital forensic investigations.

The study also highlights that even though the system was compromised, the availability of logs allowed investigators to trace back the attack patterns. This reinforces the role of logging in both detecting and responding to cyber threats.

Furthermore, the research shows that vulnerabilities in web applications can lead to serious consequences, including data breaches and unauthorized system control. Therefore, implementing strong security measures such as input validation, prepared statements, and proper access control is essential. Overall, this study underlines the importance of secure web application development and the role of forensic analysis in understanding and mitigating cyber attacks. It provides practical evidence that prevention and detection mechanisms must work together to ensure system security.

B. Conclusion

This study successfully demonstrated the forensic reconstruction of web application attacks in a controlled environment. A vulnerable web application was developed using PHP and MySQL, incorporating basic functionalities such as user authentication and CRUD operations. The application was intentionally designed with security weaknesses to simulate real-world vulnerabilities.Various controlled attacks, including SQL Injection, authentication bypass, and Insecure Direct Object Reference (IDOR), were performed on the application. The results showed that the absence of proper input validation, weak authentication mechanisms, and lack of access control can lead to serious security breaches.

During the execution of these attacks, digital forensic artefacts were generated in the form of Apache server logs and application logs. These logs provided valuable information such as request patterns, timestamps, and user activities. By analysing these artefacts, it was possible to identify suspicious behavior and reconstruct the sequence of attacker actions.

The study highlights the importance of log analysis in digital forensic investigations. Even when a system is compromised, logs serve as critical evidence for tracing attacker behavior and understanding how vulnerabilities are exploited.Overall, the research emphasizes the need for secure coding practices, proper validation mechanisms, and effective logging systems to enhance web application security and support forensic investigations.

C. Limitations

Although the study achieved its objectives, certain limitations were observed:

- The experiment was conducted in a controlled local environment and does not fully represent real-world large-scale systems.
- Only selected attack types such as SQL Injection, authentication bypass, and IDOR were considered.
- Advanced attack techniques and automated intrusion detection systems were not included in this study.
- The analysis was primarily based on Apache logs and basic application logs, without integrating advanced forensic tools.

D. Future Scope

The study can be extended further in several ways:

- Implementation of advanced attack detection techniques using machine learning and artificial intelligence.
- Integration of real-time intrusion detection and prevention systems.
- Analysis of additional forensic artefacts such as network traffic and database logs.
- Development of secure web applications with proper defense mechanisms and comparison with vulnerable systems.
- Expansion of the study to cloud-based environments and large-scale web applications.

E. Final Outcome

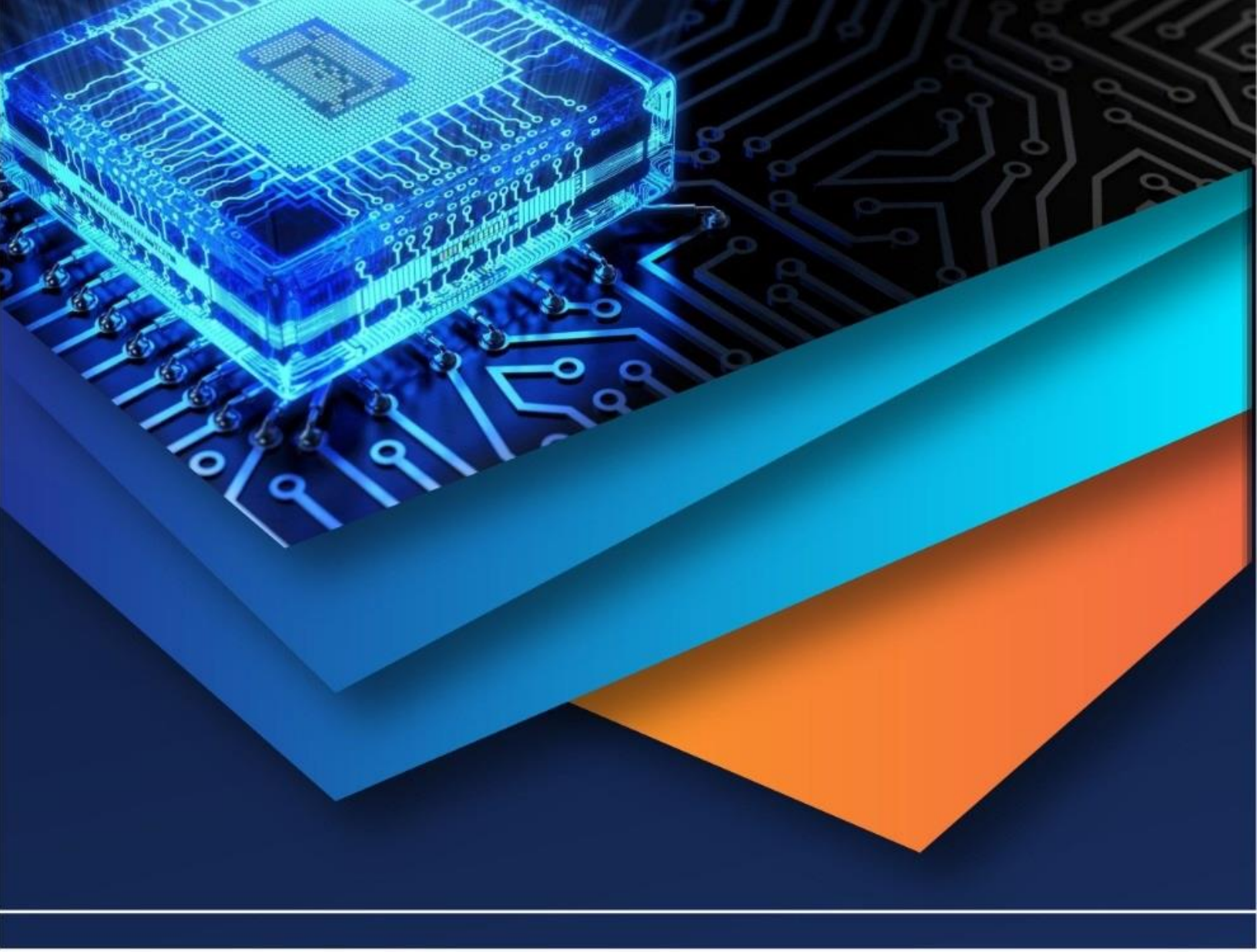
The project successfully achieved its goal of demonstrating how web application attacks can be simulated and analysed using forensic techniques. The findings provide practical insights into both cybersecurity vulnerabilities and digital forensic investigation methods.

This study serves as a foundation for further research in web security and highlights the importance of combining attack simulation with forensic analysis for better understanding and prevention of cyber threats.

REFERENCES

[1] Kumar, L., & Srivastava, P. (2025). SQL injection: A comprehensive review of methods and future directions. International Journal of Scientific Research in Science and Technology, 12(3), 979–985.
[2] Mutedi, A., & Tjahjono, B. (2022). Systematic literature review: Preventing SQL injection attacks using OWASP CSR web application firewall. Jurnal Informatika Universitas Pamulang.
[3] Bakır, R. (2025). UniEmbed: Detecting XSS and SQL injection attacks using machine learning techniques. Arabian Journal for Science and Engineering.
[4] Chen, Y., Liang, G., & Wang, Q. (2025). Research on SQL injection detection technology based on content matching and deep learning. Computers, Materials & Continua, 84(1), 1145–1167.
[5] Al Salmawi, H. M. A. (2025). Critical evaluation of SQL injection security measures in web applications. Wasit Journal for Pure Sciences, 4(1), 104–119.
[6] Purbawa, D. P., et al. (2022). An enhanced SQL injection detection using ensemble method. Jurnal Ilmiah Teknologi Informasi.
[7] Kritikos, K., et al. (2021). SQL injection detection using deep learning techniques. Journal of Physics: Conference Series.
[8] Liu, M., Li, K., & Chen, T. (2020). DeepSQLi: Deep semantic learning for testing SQL injection. arXiv preprint.
[9] Dasari, N. S., et al. (2025). Enhancing SQL injection detection and prevention using generative models. arXiv preprint.
[10] Yu, H., et al. (2026). Multi-agent honeypot-based dataset for SQL injection detection. arXiv preprint.
[11] Perera, I., et al. (2025). Detecting malicious queries using machine learning and NLP techniques. arXiv preprint.
[12] Hu, J., Zhao, W., & Cui, Y. (2020). A survey on SQL injection attacks, detection and prevention. In International Conference on Machine Learning and Computing.
[13] Aliero, M. S., et al. (2020). Detection of SQL injection vulnerability in web applications. Concurrency and Computation: Practice and Experience.
[14] Zhang, X., et al. (2021). Deep learning-based detection of web attacks. IEEE Access.
[15] Ahmed, M., et al. (2022). CNN-based detection of SQL injection attacks. Journal of Cybersecurity.
[16] Verma, R., et al. (2023). Real-time detection of web application attacks using machine learning. IEEE Transactions on Information Forensics and Security.
[17] Singh, A., et al. (2022). Log-based intrusion detection for web applications. International Journal of Computer Applications.
[18] Kumar, S., et al. (2021). Analysis of Apache logs for cyber attack detection. Procedia Computer Science.
[19] Gupta, P., et al. (2021). Client-side detection of cross-site scripting attacks. Journal of Web Engineering.

[20] Singh, R., et al. (2022). Detection of XSS attacks using NLP techniques. International Journal of Information Security.
[21] Tadhani, M., et al. (2024). Deep learning-based detection of web vulnerabilities. IEEE Access.
[22] Babaey, M., et al. (2025). AI-based framework for web attack detection. Journal of Cybersecurity.
[23] Zawoad, S., & Hasan, R. (2020). Digital forensics for web applications. IEEE Security & Privacy.
[24] Chen, X., et al. (2023). Forensic investigation of web attacks using log data. Digital Investigation.
[25] Singh, K., et al. (2022). Web log analysis for intrusion detection systems. International Journal of Digital Crime and Forensics.
[26] Perera, D., et al. (2025). AI-based forensic analysis of web logs. Journal of Cyber Forensics.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY