

Log Visualization and Access Control for Apache Webserver Intrusion Detection Decision Support

Paul Ndudiri Ohia, Pro. P.O Asagba, Emmanuel Waribo Otit

1, Department of Computer Science, University of Port Harcourt Choba, Nigeria

2,3, Department of Computer Science, University of Port Harcourt Choba, Nigeria

ABSTRACT

Webserver administrators often depend on raw access logs and error logs to understand intrusion attempts, visitor behaviour, server activity, and abnormal request patterns. However, raw logs are difficult to interpret quickly, especially when attacks come from proxies, bots, spam sources, and repeated login or injection attempts. This completed work developed a webserver log visualization and access-control decision-support system for Apache webserver intrusion detection. The developed system converts server log records into a graphical dashboard that displays detected threats, visitor analytics, host information, location patterns, device/browser information, and notification records. Access control was integrated so that the dashboard supports action, not only observation; administrators can use visual evidence to restrict suspicious IP addresses, proxy traffic, user roles, or sensitive application resources. The system was tested with detected threat categories including SQL injection, brute force, spam, proxy traffic, and bots. The results showed that proxy activity recorded the highest number of attempts, followed by spam and SQL injection. The implemented dashboard provides screenshot-ready evidence for security reporting and improves the movement from intrusion detection to administrator decision-making.

Keywords: Webserver Logs, Log Visualization, Access Control, Intrusion Detection, Decision Support, Apache, Security Dashboard

1. INTRODUCTION

Webserver application environments have become essential platforms for delivering public, commercial, educational, financial, and organizational services. Modern organizations increasingly depend on web-based applications to provide users with continuous access to information, transactions, communication services, administrative functions, and other digital resources. As dependence on these applications increases, webserver have also become attractive targets for unauthorized access, automated attacks, credential abuse, malicious requests, and other forms of intrusion. Consequently, effective monitoring and control of activities occurring within webserver environments have become important components of application and information-system security.

Every interaction with a webserver can generate information that is useful for security monitoring. Normal page visits, HTTP requests, successful and failed login attempts, access denials, administrative activities, bot requests, suspicious URLs, repeated connection attempts, and potential attacks may leave traces in server log files. These records can provide valuable evidence about who accessed the server, what resources were requested, when the activity occurred, where the request originated, and how the server responded. When properly analysed, such information can assist administrators in identifying abnormal behaviour, tracing suspicious activities, investigating security incidents, and determining appropriate security responses.

Despite their importance, webserver logs are commonly stored as large volumes of structured or semi-structured textual records. As the number of users and requests increases, manually reading and interpreting these records becomes increasingly difficult and time-consuming. Important security events may be distributed across thousands of log entries, making it challenging for an administrator to distinguish legitimate activities from suspicious behaviour within a short period. Raw log records alone may therefore provide insufficient situational awareness, particularly when rapid security decisions are required. This creates a need for mechanisms that can transform webserver log data into meaningful and easily interpretable security information.

Log visualization provides an important approach for addressing this challenge. Rather than requiring administrators to continuously examine raw log files, relevant records can be processed, categorized, summarized, and presented through a graphical monitoring interface

2. RELATED WORK

Current web application security guidance emphasizes both visibility and control. OWASP (2021a) identifies broken access control as a major risk because users may bypass restrictions and reach resources outside their permission level. OWASP (2021b) also identifies logging and monitoring failures as a risk because attacks may continue when security events are not captured, escalated, and reviewed in time.

NIST's Zero Trust Architecture (Rose et al., 2020) strengthens the same idea by recommending continuous evaluation of access requests rather than broad trust based

on network position. For webserver dashboards, this means logs should be interpreted in relation to access decisions: who requested the resource, from where, under what role, and with what behaviour.

Recent intrusion detection studies also support log-based and visualization-based decision support. Kim et al. (2020) proposed AI-IDS for real-time web intrusion detection, while Farooq and Anjum (2020) studied webserver attack detection using machine learning. Zhong (2021) presented security log analysis for IDS warning generation. In addition, visual analytics research in 2020 and 2021 shows that security dashboards can help analysts interpret large volumes of events more quickly than raw log reading.

The main chapter work reviewed systems that detect attacks but noted that administrators still need a clearer interface for analysing logs and responding to detected threats. This completed work demonstrates how log visualization and access control worked together as a decision-support mechanism.

3. CHALLENGES OF THE EXISTING SYSTEM

The existing webserver security process had several weaknesses that motivated the development of a visualization and access-control dashboard. These challenges are described below.

- I. Raw log dependency: administrators had to read access logs and error logs manually, which is slow and difficult during repeated attacks.
- II. Poor visibility of attack categories: intrusion attempts such as SQL injection, brute force, spam, proxy traffic, and bots were not always summarized in a way that supported quick decisions.
- III. Delayed response: when detected events were only stored in logs, the administrator could miss urgent incidents that required blocking or restriction.
- IV. Weak connection between detection and access control: detected suspicious behaviour did not always lead to a clear access-control action such as blocking an IP address, restricting a role, or denying a protected route.
- V. Limited geographical and visitor analytics: without location and visitor summaries, it was harder to understand attack sources and recurring request patterns.
- VI. Lack of screenshot-ready reporting: the system needed clear screens that could be captured and used for publication, audit, and administrative reporting.

Existing Challenge	Visualization and Access-Control Improvement
Manual log reading	Dashboard summarizes detected logs, visitors, and attack categories.
Delayed response	Alerts and visual indicators support quicker administrator action.
Detection without enforcement	Access-control decisions restrict suspicious sources and roles.

Hard-to-read attack sources	Location and visitor analytics show repeated suspicious origins.
Poor reporting evidence	Screenshots can document dashboard, detected records, and alert views.

4. DESIGN METHODOLOGY

This study adopted a Design Science Research Methodology (DSRM) combined with a design and development approach. The methodology was considered appropriate because the research focuses on identifying limitations in an existing web-server intrusion-detection process and developing a practical technological solution to address them.

4.1 IMPLEMENTED SYSTEM ARCHITECTURE

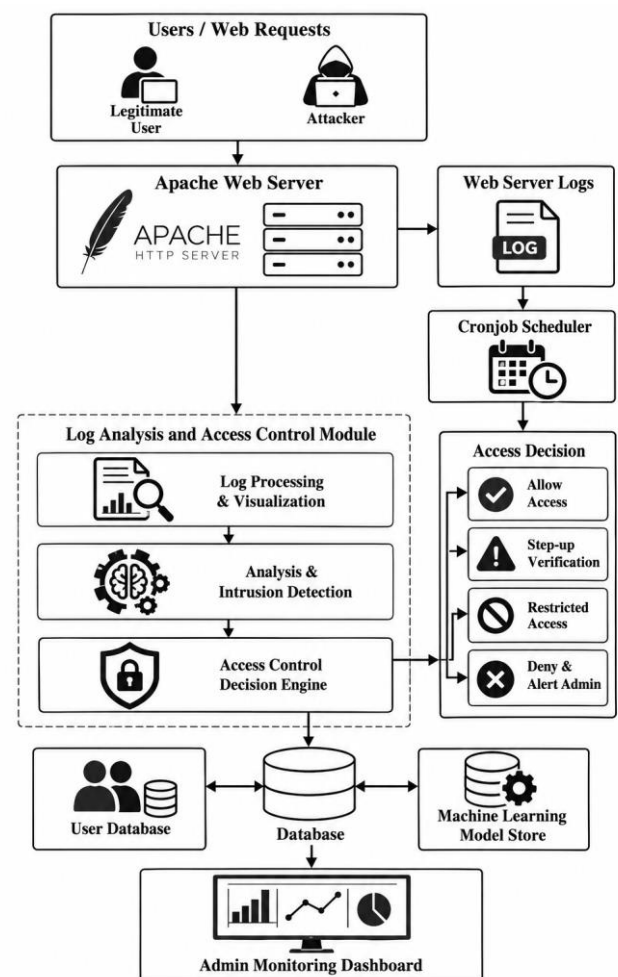


Figure 1: Implemented system architecture.

Figure 1 presents the implemented architecture of the system. It shows how Apache webserver requests move from log generation to cronjob processing, visualization, intrusion analysis, access-control decisions, alerting, and administrator review. The implemented architecture provides a web-server log visualization and access-control framework for intrusion detection decision support. User requests are processed by the Apache web server, while generated logs are collected and periodically processed through the Cronjob Scheduler.

The Log Processing and Visualization Module extracts and presents relevant activities, which are subsequently evaluated by the Analysis and Intrusion Detection Module to identify suspicious behaviour or potential attacks.

4.2 Data Collection and Log Sources

The system collects data from Apache access logs, error logs, IDS scripts, visitor activity, host information, and detected attack categories. Relevant fields include request source, user agent, operating system, device type, country/location, time of request, attack category, proxy status, and response action.

4.3 Dashboard and Decision-Support Design

The dashboard presents log data in a structured interface. Instead of forcing the administrator to inspect raw log lines, the system shows detected records, threat categories, visitor analytics, host details, storage information, and communication alerts. This supports faster understanding of security events.

The access-control decision layer uses dashboard evidence to recommend or enforce restrictions. For example, repeated proxy traffic can be blocked, suspicious IP addresses can be banned, abnormal login attempts can be restricted, and sensitive pages can require stronger authentication.

4.4 Operational Workflow

Step 1: Apache webserver records user and automated requests. Step 2: IDS scripts classify suspicious events. Step 3: Detected events are stored in the database. Step 4: The dashboard visualizes the records as tables, charts, and summaries. Step 5: The administrator reviews the visual evidence. Step 6: Access-control actions are applied to restrict suspicious sources, roles, or protected resources. Step 7: Alerts and reports are generated for follow-up.

5. RESULTS AND DISCUSSION

The source results show that several threat types were captured during deployment and testing. These include SQL injection, brute force attempts, spam requests, proxy traffic, and bot activity. Presenting these categories visually makes them easier to compare and respond to.

Detected Threat	Total Attempts	Dashboard Interpretation	Access-Control Decision
SQL Injection	4	Input pattern indicates attempt to manipulate database query.	Reject request and flag source.
Brute Force	1	Repeated login pressure on authentication interface.	Restrict account/IP and alert admin.
Spam	10	Repeated abnormal form or request behaviour.	Limit source activity.
Proxy	18	Suspicious masked traffic with repeated requests.	Block or challenge proxy source.
Bots	2	Automated request behaviour detected.	Restrict repeated automated access.

Proxy attempts were the highest category, followed by spam and SQL injection. This finding supports the need for visual log analysis because the administrator can

immediately see which threat category dominates and which access-control response should be prioritized.

Location	Number of Records
United States	10
Russia	3
Nigeria	6
Netherlands	2
Ireland	2
India	1
Thailand	2
Germany	1
France	4
United Kingdom	1

The location summary also supports decision-making. It does not prove that every request from a country is malicious, but it helps the administrator identify repeated traffic patterns that may require closer review, rate limiting, or access-control restriction.

IP Address	Date	Browser	OS	Country	Actions
41.190.3.195	04 August 2021 at 15:58	Google Chrome 92.0.4515.107	Windows 7 x64	Nigeria	[Details] [Delete]
41.190.3.195	04 August 2021 at 15:16	Google Chrome 79.0.3945.116	Android 9	Nigeria	[Details] [Delete]
41.190.30.22	04 August 2021 at 11:24	Google Chrome 92.0.4515.107	Windows 7 x64	Nigeria	[Details] [Delete]

Figure 2: Detected access logs summary screen.

Figure 2 shows the detected access logs summary screen. It provides a quick view of captured suspicious activities and gives the administrator a first point of reference before opening the detailed dashboard records.

Address	Type	Date	Browser	OS	Country	Actions
5.154.61.206	Proxy	05 August 2021 at 03:57	Firefox 58.0	Ubuntu x64	France	[Details] [Delete]
2.83.146.233	Proxy	05 August 2021 at 03:46	Firefox 58.0	Ubuntu x64	France	[Details] [Delete]
108.182.5	Spammer	04 August 2021 at 23:43	Google Chrome 90.0.4430.72	Windows 10 x64	United States	[Details] [Delete]
3.115.200.170	Spammer	04 August 2021 at 17:09	Google Chrome 78.0.3904.108	GNU/Linux x64	India	[Details] [Delete]
96.980:3600	Missing User-Agent header	04 August 2021 at 17:09	Unknown		Unknown	[Details] [Delete]
79.149.177	Spammer	04 August 2021 at 17:08	Google Chrome 78.0.3904.108	GNU/Linux x64	Thailand	[Details] [Delete]

Figure 3: Detected logs dashboard.

Figure 3 shows the detected logs dashboard. The screen organizes suspicious webserver records into readable entries so the administrator can identify attack type, source information, and response needs without manually reading raw log files.



Figure 4: Device and browser agent access visualization chart.

Figure 4 explains how the system visualizes browser, device, and user-agent information. This helps the administrator recognize repeated automated requests, unusual clients, and access patterns that may support blocking or restriction decisions.

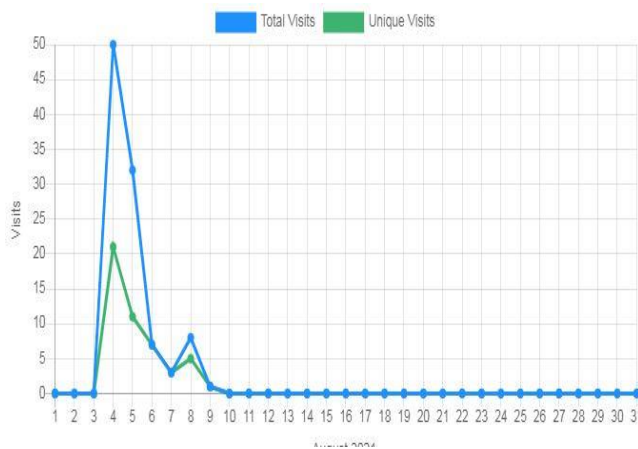


Figure 5: Visitor analytics visualization chart.

Figure 5 presents visitor analytics generated from the collected logs. The chart supports quick comparison of normal and suspicious activity trends and helps the administrator understand the volume and behaviour of requests over time.

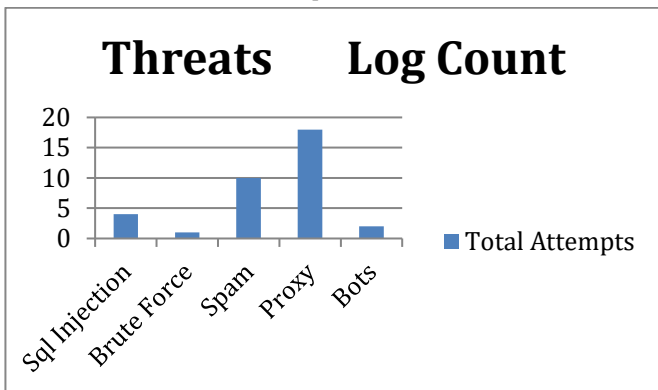


Figure 6: Location, alert, and access-control action screen.

Figure 6 shows the alert and access-control evidence area. It connects detected events with administrative action by highlighting suspicious sources that may require blocking, restriction, review, or stronger authentication.

VISITORS BY LOCATION

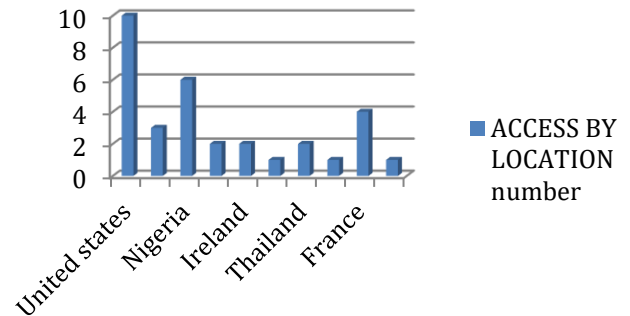


Figure 7: Visitors access by location visualization chart.

Figure 7 presents location-based visitor analysis. The visualization does not automatically classify a country as malicious, but it helps identify repeated traffic origins that deserve further inspection or rate limiting.

6. SUMMARY

This paper presented the completed log visualization contribution from the main webserver intrusion detection work. It explained why raw logs are difficult to use, how dashboard visualization improved security interpretation, and how access control turned the dashboard into a decision-support tool.

7. CONTRIBUTION TO KNOWLEDGE

This study contributes to knowledge by showing how Apache webserver logs can be transformed from raw text records into visual security intelligence that supports access-control decision-making. The work demonstrates that intrusion detection becomes more practical when detected events are presented through dashboards, charts, visitor summaries, location views, and alert records.

The study also contributes a practical link between log visualization and access control. Instead of stopping at detection, the developed system supports decisions such as blocking suspicious sources, reviewing abnormal traffic, restricting risky requests, and protecting sensitive resources. This strengthens the role of visualization as an enforcement-support tool for webserver security administration.

8. CONCLUSION

Webserver log visualization improves intrusion detection because it converts raw records into understandable security information. When access control is added, the administrator gains a practical path from observation to action. The system can show suspicious behaviour, explain the category of threat, preserve evidence, and support restrictions against risky sources, roles, or protected resources.

9. RECOMMENDATIONS

- I. Webserver administrators should use dashboards to summarize intrusion logs instead of relying only on raw text log files.
- II. Detected log screens should include access-control actions such as block, restrict, review, or require stronger authentication.
- III. Visitor analytics and location views should be used carefully as decision-support evidence, not as the only basis for blocking.
- IV. Screenshots of detected records, visitor analytics, and alert pages should be added to the final paper as system evidence.
- V. Further work should test the dashboard with larger log datasets and automated recommendation rules for access-control decisions.

REFERENCES

- [1]. Bach-Nutman, M. (2020). Understanding the Top 10 OWASP vulnerabilities. arXiv preprint.
- [2]. Farooq, M. S., and Anjum, S. (2020). Web Server Attack Detection using Machine Learning. International Conference on Cyber Warfare and Security.
<https://doi.org/10.1109/ICCWS48432.2020.9292393>
- [3]. Kim, A. C., Park, M., and Lee, D. H. (2020). AI-IDS: Application of Deep Learning to Real-Time Web Intrusion Detection. IEEE Access, 8, 70245-70261.
<https://doi.org/10.1109/ACCESS.2020.2986882>
- [4]. OWASP Foundation. (2021a). A01:2021 - Broken Access Control. OWASP Top 10.
https://owasp.org/Top10/A01_2021-Broken_Access_Control/
- [5]. OWASP Foundation. (2021b). A09:2021 - Security Logging and Monitoring Failures. OWASP Top 10.
https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/
- [6]. OWASP Foundation. (2021c). OWASP Top 10:2021. <https://owasp.org/Top10/>
- [7]. Rose, S., Borchert, O., Mitchell, S., and Connelly, S. (2020). Zero Trust Architecture. NIST Special Publication 800-207.
<https://doi.org/10.6028/NIST.SP.800-207>
- [8]. Zhong, W. (2021). A Security Log Analysis Scheme Using Deep Learning Algorithm for IDSs in Social Network. Security and Communication Networks, 2021, Article ID 5542543.
<https://doi.org/10.1155/2021/5542543>