Design And Development of A Hybrid Web Application Firewall For Enhanced Web Security

Darshan Karkar[1], Prof. Shweta Katariya[2]
[1]Dept of Computer Engineering
[2]Assist.Professor, Dept of Computer Engineering
[1, 2] Dr. Subhash University, Junagadh

Abstract- *The increasing sophistication of web-based attacks such as SQL Injection, Cross-Site Scripting (XSS), and emerging zero-day threats has necessitated the evolution of intelligent and adaptive defence mechanisms. This research presents the design and development of a hybrid Web Application Firewall that integrates both signature-based and anomaly-based detection techniques to enhance web application security. The proposed system leverages machine learning algorithms for anomaly detection and pattern-matching methods for known attack signatures, achieving a balanced trade-off between accuracy and efficiency. The architecture incorporates real-time traffic monitoring, feature extraction, and multi-layer threat analysis, enabling proactive defence against evolving attack patterns. Experimental evaluation demonstrates that the hybrid WAF significantly improves detection accuracy while reducing false positives compared to traditional approaches. The system also exhibits scalability and adaptability for modern web environments. This study contributes to the advancement of intelligent WAF architectures capable of mitigating both known and unknown web threats, thereby strengthening overall cybersecurity resilience.*

I. INTRODUCTION

The rapid expansion of web technologies has increased the vulnerability of web applications to attacks such as SQL Injection (SQLi), Cross-Site Scripting (XSS), and other evolving cyber threats. These exploits threaten data integrity, confidentiality, and organizational trust. To address these challenges, this research presents the design and development of a hybrid Web Application Firewall that combines traditional and intelligent detection techniques for enhanced web security.[1,3]

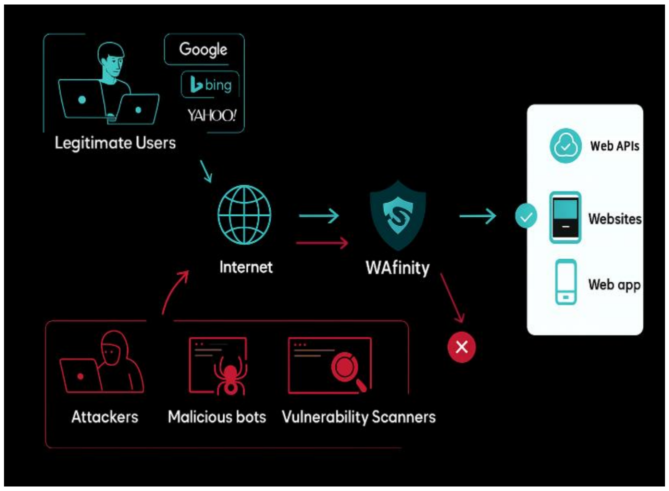

Image 1.1: Process Flow [2]

The proposed system filters and monitors HTTP/HTTPS traffic between clients and web servers, detecting and blocking malicious requests in real time. Unlike conventional WAFs that rely solely on signature-based detection, the hybrid model integrates signature-based rules for identifying known attacks with machine learning-driven anomaly detection to uncover obfuscated and zero-day threats. Through traffic pattern analysis and behavioural deviation monitoring, the system adapts dynamically to emerging attack vectors. [1,2]

Experimental evaluations demonstrate that the hybrid WAF achieves improved detection accuracy and reduced false positives compared to traditional methods. By leveraging adaptive learning and multi-layered defence strategies, the framework provides a proactive, scalable, and resilient solution for safeguarding modern web applications against both known and unknown threats. [4]

II. RESEARCH OBJECTIVES

The research aims to develop a hybrid Web Application Firewall that integrates rule-based, anomaly-based, and adaptive machine learning techniques for effective detection of SQLi, XSS, and emerging web threats. It further focuses on implementing real-time monitoring, logging, and adaptive filtering mechanisms to enhance detection accuracy and system responsiveness. The proposed model will be evaluated against existing WAFs using real-world datasets to demonstrate improved performance, scalability, and efficiency.[2,5]

1. Analyze existing Web Application Framework techniques and identify gaps in detecting SQLi, XSS, and emerging threats. [1]
2. Design a hybrid detection engine integrating rule-based, anomaly-based, and adaptive learning methods. [5]
3. Implement SQLi and XSS prevention modules with real-time filtering and adaptive learning. [6]
4. Develop a logging and monitoring interface for real-time attack visualization and alerts.[7]
5. Evaluate system performance using real-world datasets and penetration testing tools.[8]
6. Compare detection accuracy, false positives, and efficiency with existing Web Application Frameworks [3].
7. Ensure scalability and modular architecture for cloud and enterprise integration.[9]

III. RESEARCH GAP

Despite the availability of commercial and open-source Web Application Firewalls, several critical challenges remain:

High False Positives/Negatives – Many existing WAFs either block legitimate traffic or fail to detect sophisticated attacks.[1,2]

Lack of Adaptability – Signature-based systems cannot effectively detect zero-day or obfuscated attacks, while anomaly-based systems often lack precision.[3]

Limited Hybridization – Few WAFs combine signature-based, anomaly-based, and heuristic/ML-based detection into a single adaptive system.[5]

Scalability Issues – Traditional WAFs struggle with integration into cloud-native and large-scale enterprise systems, leading to performance degradation.[8]

Weak Monitoring and Usability – Current systems often lack intuitive dashboards, making real-time monitoring, logging, and analysis difficult for administrators. [9,10]

IV. LITERATURE REVIEW

1) Classical WAFs & Signature-Based Detection

Early and widely deployed WAFs (e.g., reverse-proxy or in-app modules) rely on rule/signature matching to detect known attack patterns (regexes, token blacklists, request normalization plus matching). Community rule sets (e.g., generic CRS-style approaches) popularized portable signatures for common payloads (SQLi tautologies, UNION/SELECT patterns, <script> tags, event handlers).

Strengths: low latency, deterministic behavior, explainability.
Limitations: brittle to obfuscation (URL/Unicode/CASE/COMMENT tricks), high maintenance cost, and poor coverage of zero-day or polymorphic attacks.

2) Anomaly-Based Detection (Statistical & Unsupervised)
Second-generation systems model “normal” traffic and flag deviations. Techniques include:

Payload statistics: length distributions, parameter entropy, character n-grams, token frequency.

Sequence modeling: Markov chains over URL/verb/parameter sequences; time-series baselining for user/session behaviors.
Unsupervised learners: One-Class SVM, Isolation Forest, autoencoders, PCA for dimensionality reduction.

Pros: detection of novel patterns.
Cons:calibration and drift issues; susceptibility to higher false positives on genuine traffic changes (new features, campaigns, A/B tests).

3) Machine Learning & Deep Learning for Web Attack Detection

Research has accelerated on supervised ML over HTTP features (header fields, query bodies, DOM signals) using SVM, Random Forest, Gradient Boosting/XGBoost; and DL over raw or lightly processed payloads.

Feature-engineered ML: token counts, SQL keywords, special-char ratios, DOM event tokens, encoder presence (Base64), and request structural features often outperform naive regex.

DL architectures: CNNs over character streams; Bi-LSTM/GRU for sequence patterns; hybrids (CNN+LSTM) for local+long-range features; transformer-based models for complex context and multi-parameter relations.
Key findings: ML/DL improves recall for obfuscated SQLi/XSS, but needs careful class imbalance handling, cost-sensitive training, and explainability (e.g., SHAP, LIME) for operator trust.

4) Hybrid Detection (Rules + Anomaly/ML)

“Hybrid” combines deterministic rules for high-confidence signatures with ML/anomaly layers for edge cases and zero-days. Common designs:

Two-stage gates: fast rules first; uncertain traffic escalates to ML/DL.

Ensembles: weighted voting of rule score, anomaly score, and classifier probability; stacking for meta-decisions.

Active/adaptive learning: feedback from analysts to refine thresholds/rules; periodic retraining; drift detection.
Outcome in literature: lower false positives than pure anomaly methods and better zero-day capture than pure signatures—at modest added latency if engineered carefully.

5) SQL Injection (SQLi) Mitigation

Beyond WAFs, program-level defenses include prepared statements/parameterized queries, taint tracking, and static/dynamic analysis of query construction. WAF-focused studies highlight:

Robust lexer/parser-aware detection (e.g., checking grammar conformity of SQL segments).

Encoding/normalization pipelines to defeat comment/whitespace and encoding-based evasions.

ML over features like keyword proximity, operator frequency, and syntax anomalies.
Trend: hybrid WAFs use grammar-aware prefilters + ML classifiers for mutated payloads.

6) Cross-Site Scripting (XSS) Mitigation

At the app layer, context-aware output encoding and CSP (Content Security Policy) are proven. For WAFs:

Rules for <script>, event handlers, javascript: URIs, srcdoc, SVG/MathML vectors, and DOM sinks.

ML/DL approaches learn patterns of script injection, attribute abuse, and DOM-based XSS indicators.
Modern evasion: nested encodings, template injections, JSON/JS-in-JSON, and inline event/URL handler variants.

7) Emerging Web Threats Relevant to WAFs

Recent literature broadens beyond SQLi/XSS to:

SSRF, Server-Side Template Injection (SSTI), Deserialization flaws, RCE via gadget chains, GraphQL/API abuse, NoSQL injection, JWT attacks, HTTP smuggling, and supply-chain (NPM/PyPI) vectors.

Cloud-native vectors: metadata service access, misconfigured API gateways, serverless cold starts exploited for timing, and service mesh misconfigurations.
Hybrid WAFs increasingly include protocol/state checks, API-aware schemas, and schema validation (e.g., OpenAPI/GraphQL introspection controls).

V. PROPOSED SYSTEM

1. Start

- This marks the beginning of the HTTP traffic monitoring process.
- The system is initialized and ready to handle incoming traffic.

2. Incoming Traffic

- Web clients (browsers, APIs, bots, etc.) send HTTP requests to the server.
- These requests are intercepted before reaching the web application.

3. Detection Engine

- The intercepted HTTP request is forwarded to the Detection Engine.
- This component is responsible for analysing the request to determine if it contains malicious patterns or behaviours.

The detection engine typically uses:

- Signature-based rules (known attack patterns)
- Heuristic or anomaly-based analysis (unusual or suspicious behavior)
- Machine learning algorithms (for unknown or zero-day attacks)

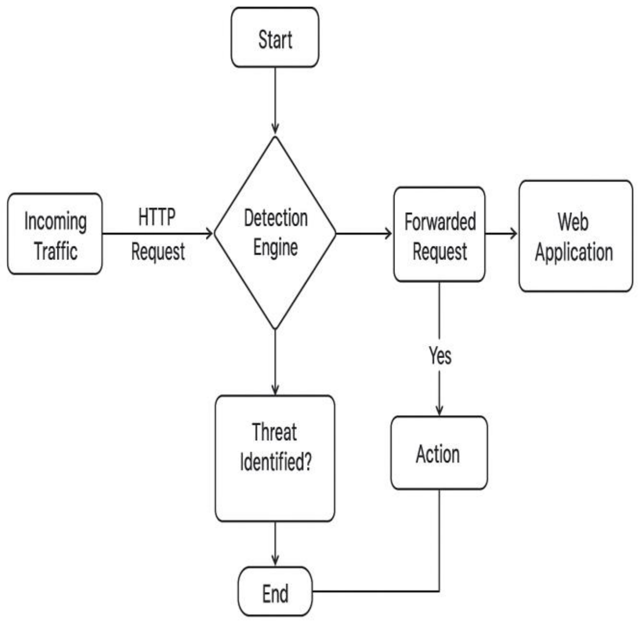

Image 2: Proposed System

4. Decision Point: Threat Identified

The engine evaluates the request and answers: Is there a threat present?

Two possibilities arise:

a. No Threat Detected

The request is considered safe.
It is forwarded to the next step: "Forwarded Request".

b. Threat Detected

The request is malicious or suspicious.
It is passed to the Action module for mitigation.

5. Forwarded Request

- If the request is clean, it is sent directly to the web application backend.
- The web app then processes it normally and sends a response to the client.

6. Web Application

- Represents the protected application (e.g., a website, API, or service).
- At this point, it processes only those requests that are deemed safe.

7. Action (if threat detected)

When a threat is identified, the system takes a defensive action. This may include:

- Blocking the request entirely
- Redirecting the request to a warning page
- Sanitizing the input to remove malicious code
- Logging and alerting the security team
- Rate limiting or banning IP addresses in case of repeated attacks

8. End

- The process completes for the current HTTP request.
- This either happens after a clean request is served or after a malicious one is blocked or mitigated.

VI. CONCLUSION& FUTURE WORK

This research work provides modern, adaptive protection against a wide range of web threats, including SQL Injection, XSS, CSRF, and zero-day attacks. Unlike traditional WAFs that rely only on signatures, it uses a hybrid detection approach—combining rule-based filtering, anomaly detection, and machine learning—to detect both known and unknown threats. It intelligently analyses HTTP/HTTPS traffic, reduces false positives, and continuously learns from behavior patterns. With real-time threat intelligence, logging, and reporting, it ensures your web applications stay secure and resilient in an evolving threat landscape.

Future enhancements of this hybrid WAF can focus on integrating external threat intelligence for real-time updates, adopting deep learning to detect complex threats, and inspecting encrypted HTTPS traffic. Adding user behavior profiling and explainable AI can improve accuracy and transparency. Auto-remediation features and deployment in edge/cloud environments will enhance scalability and responsiveness. Additionally, improving adversarial robustness and expanding training datasets can further strengthen detection capabilities.

REFERENCES

[1] A. Shaheed and M. H. D. Bassam Kurdy, “Web Application Firewall Using Machine Learning and Features Engineering,” *Security and Communication Networks*, vol. 2022, 5280158, 2022. (Wiley Online Library)

[2] “Artificial Intelligence Web Application Firewall for advanced detection of web injection attacks,” *Expert Systems*, 2025. (Wiley Online Library)

[3] F. M. Alotaibi and V. G. Vassilakis, "Toward an SDN-Based Web Application Firewall: Defending against SQL Injection Attacks," *Future Internet*, vol. 15, no. 5, article 170, 2023. (MDPI)

[4] I. P. Arya Dharmaadi, E. Athanasopoulos, and F. Turkmen, "Fuzzing Frameworks for Server-side Web Applications: A Survey," arXiv, 2024. (arXiv)

[5] Z. Qu, X. Ling, T. Wang, X. Chen, S. Ji, and C. Wu, "AdvSQLi: Generating Adversarial SQL Injections against Real-world WAF-as-a-service," arXiv, 2024. (arXiv)

[6] L. Demetrio, A. Valenza, G. Costa, and G. Lagorio, "WAF-A-MoLE: Evading Web Application Firewalls through Adversarial Machine Learning," arXiv, 2020. (arXiv)

[7] C. Wu, J. Chen, S. Zhu, W. Feng, R. Du, and Y. Xiang, "WAFBOOSTER: Automatic Boosting of WAF Security Against Mutated Malicious Payloads," arXiv, 2025. (arXiv)

[8] B. Shah and A. Shah, "Survey on Existing Enterprise Web Application Security Mechanisms Using Machine Learning," *Journal of Electrical Systems*, vol. 20, no. 11s, 2024. (Journal of Electrical Systems)

[9] M. Surekha, K. Kiran Kumar, M. V. S. Prasanth, and P. S. G. Aruna Sri, "Web application firewall using XSS," *International Journal of Engineering and Technology*, vol. 7, no. 2.7, pp. 941–943, 2018. (Science Publishing Corporation)

[10] Y. Zhang and T. Zhang, "Research Into the Security Threat of Web Application," *Journal of Web Engineering*, vol. 5, no. 10, pp. 43-51, 2023. (River Publishers Journals)