Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

# IntelliWAF: A Six-Layer Intelligent Web Application Firewall with Machine Learning and Anomaly Detection for Real-Time Web Threat Mitigation

**Pranav Kaluram Shinde**
Department of Computer Engineering
JSPM's JSCOE, Pune

**Yash Ranjit Ingole**
Department of Computer Engineering
JSPM's JSCOE, Pune

**Neha Pintu Tikore**
Department of Computer Engineering
JSPM's JSCOE, Pune

Gaurav Sanjay Gosavi
Department of Computer Engineering
JSPM's JSCOE, Pune

**Dr. S.B. Chaudhari**
Department of Computer Engineering
JSPM's JSCOE, Pune

*Abstract* - **Abstract - Web application security has become a critical concern as modern applications face increasingly sophisticated and evolving attack vectors that traditional signature-based firewalls consistently fail to address. Existing Web Application Firewalls (WAFs) rely heavily on static rule databases that require frequent manual updates, lack the ability to detect zero-day threats, and often demand significant infrastructure changes to the protected application. This paper presents IntelliWAF, a six-layer intelligent Web Application Firewall designed to overcome these limitations through a combination of regex-based rule matching, machine learning anomaly detection, and asynchronous AI-assisted semantic analysis, all without requiring any modification to the protected web application. IntelliWAF operates as a transparent reverse proxy built on mitmproxy and Nginx, containerized entirely using Docker for platform-independent deployment. The detection pipeline consists of six sequential layers: IP blocklisting, sliding window rate limiting, 24-rule regex pattern matching, Random Forest classification, Isolation Forest anomaly scoring, and gray-zone semantic analysis. The Random Forest model, trained on the CSIC 2010 HTTP dataset comprising 61,065 samples, achieves a classification accuracy of 91.5% across attack categories including SQL injection, cross-site scripting, path traversal, SSRF, and command injection. A Redis-backed event queue decouples traffic processing from database logging, enabling real-time attack visibility through a Socket.IO-powered dashboard with sub-50ms event latency. Evaluation results demonstrate that IntelliWAF effectively detects both known and obfuscated attack patterns while supporting dynamic multi-site management without service interruption. Keywords — Web Application Firewall, Intrusion Detection, Machine Learning, Random Forest, Isolation Forest, Anomaly Detection, mitmproxy, SQL Injection, Cross-Site Scripting, Redis, Docker,**

**Real-Time Threat Detection, CSIC 2010, Reverse Proxy, Network Security**

**Keywords – Web Application Firewall; Machine Learning; Random Forest; Isolation Forest; Anomaly Detection; Large Language Models; SQL Injection; Cross-Site Scripting; Server-Side Request Forgery; Prompt Injection; Log4Shell; OWASP; Reverse Proxy; Cybersecurity**

## I. INTRODUCTION

Web applications have become the primary platform for delivering digital services across domains such as banking, healthcare, education, e-commerce, and government systems. As organizations increasingly depend on web-based infrastructures, the attack surface exposed to the Internet continues to expand. Vulnerabilities in web applications are frequently exploited through attacks such as SQL Injection (SQLi), Cross-Site Scripting (XSS), Server-Side Request Forgery (SSRF), command injection, and path traversal. These attack vectors remain among the most significant security concerns identified by the OWASP Top Ten project, highlighting the need for effective protection mechanisms for Internet-facing applications [1].

Web Application Firewalls (WAFs) are commonly deployed as a defensive layer between clients and web servers to inspect incoming HTTP requests and block malicious traffic before it reaches the application. Traditional WAF solutions, including rule-based systems such as ModSecurity, primarily rely on predefined signatures and manually maintained rule sets [11]. While such approaches are effective against known attack patterns, they often struggle to identify previously unseen threats, heavily obfuscated payloads, or attacks that intentionally evade static signature matching. As web attacks

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

continue to evolve, maintaining comprehensive and up-to-date rule databases becomes increasingly challenging [10].

To address these limitations, recent research has explored the application of machine learning techniques for web attack detection. Random Forest models have demonstrated strong classification performance in intrusion detection environments due to their ability to learn complex relationships from traffic features while maintaining robustness against overfitting [4], [5]. Similarly, anomaly detection approaches based on Isolation Forest have shown promise in identifying unusual or previously unseen behaviors without requiring explicit attack signatures [3], [14]. Research on anomaly-based WAF architectures further suggests that combining statistical analysis with HTTP-specific features can improve the detection of sophisticated attack attempts that may bypass conventional rule-based defenses [8].

Despite these advancements, many existing solutions focus on a single detection strategy and often lack the flexibility required for practical deployment in heterogeneous web environments. Systems based solely on signatures may miss novel threats, whereas purely anomaly-based approaches can generate excessive false positives. Furthermore, integrating advanced detection mechanisms into existing infrastructures frequently requires significant configuration effort and operational overhead.

To overcome these challenges, this paper presents **IntelliWAF**, a six-layer intelligent Web Application Firewall designed for real-time web threat mitigation. IntelliWAF combines multiple defensive mechanisms, including IP reputation filtering, rate limiting, signature-based detection, machine learning classification using Random Forest, and anomaly detection using Isolation Forest within a unified reverse-proxy architecture. The system is implemented using Nginx, mitmproxy, Redis, MySQL, Docker, and Scikit-learn, enabling transparent deployment without requiring modifications to the protected web application. By combining deterministic rule matching with machine learning-based analysis, IntelliWAF aims to improve detection accuracy while maintaining operational simplicity and real-time visibility into web application threats.

## II. RELATED WORK

Web Application Firewalls (WAFs) have traditionally served as the first line of defense against web-based attacks by inspecting HTTP requests and blocking malicious traffic before it reaches the target application. Most commercial and open-source WAF solutions rely on signature-based detection mechanisms, where predefined rules are used to identify known attack patterns. While these approaches remain effective against well-documented threats, their dependence on manually maintained rule sets limits their ability to detect previously unseen or highly obfuscated attacks. Mitropoulos et al. [10] highlighted that modern web attacks continue to evolve in complexity, creating challenges for conventional defense mechanisms that rely solely on static signatures.

One of the most widely adopted open-source WAF solutions is ModSecurity, which employs a rule-based inspection engine to identify malicious HTTP requests. Azer and Sobh [11] evaluated ModSecurity against SQL injection attacks and demonstrated its effectiveness in detecting known attack signatures. However, the study also indicated that rule-based systems require continuous updates and may be bypassed through encoding techniques or novel payload structures that are not represented within existing rule databases.

To overcome the limitations of signature-based detection, researchers have explored anomaly-based approaches that learn the normal behavior of web traffic and identify deviations from expected patterns. Torrano-Gimenez et al. [8] proposed an anomaly-based WAF using HTTP-specific features and one-class classification techniques. Their work demonstrated that anomaly detection can identify previously unseen attack behaviors without requiring explicit attack signatures. Although anomaly-based methods improve adaptability, they often suffer from increased false-positive rates when legitimate traffic deviates from learned patterns.

Machine learning techniques have gained significant attention in intrusion detection and web security research due to their ability to analyze complex feature relationships within network and application traffic. Breiman's Random Forest algorithm [4] remains one of the most widely used ensemble learning methods because of its high classification accuracy, resistance to overfitting, and ability to handle large feature spaces. Building upon these advantages, Zhang et al. [5] demonstrated that Random Forest models can effectively classify malicious traffic in intrusion detection systems, making them suitable candidates for web attack detection tasks.

Recent studies have further investigated the application of machine learning to web-specific attacks. Tang et al. [15] proposed a neural-network-based approach for SQL injection detection, while Sharma et al. [16] focused on detecting SQL Injection and Cross-Site Scripting attacks in multi-tier web applications. These studies demonstrated that machine learning models can improve attack detection performance compared to traditional signature-only systems. However, many existing solutions focus on a limited set of attack categories or require extensive feature engineering tailored to specific environments.

In addition to supervised learning techniques, anomaly detection models have emerged as an effective approach for identifying previously unknown threats. Liu et al. [3] introduced the Isolation Forest algorithm, which isolates anomalous observations rather than modeling normal behavior explicitly. Due to its computational efficiency and ability to detect outliers in high-dimensional data, Isolation Forest has been adopted in various cybersecurity applications. Mahoney et al. [14] further demonstrated the effectiveness of Isolation Forest for detecting abnormal behavior within network environments, reinforcing its suitability for identifying zero-day and low-frequency attack patterns.

More recently, researchers have explored adaptive and intelligent security systems capable of responding dynamically to evolving attack strategies. Amouei et al. [9] proposed a reinforcement-learning-driven framework for vulnerability discovery and WAF testing, illustrating the growing trend toward intelligent and adaptive security architectures. Similarly, Applebaum et al. [13] compared signature-based and machine-learning-based WAF approaches, concluding that hybrid detection architectures offer a promising balance between detection accuracy and operational practicality.

Despite these advancements, existing solutions often rely on a single detection methodology, either rule-based inspection or machine learning classification. Such approaches may either miss novel attacks or generate excessive false positives when deployed in real-world environments. Furthermore, many research prototypes focus primarily on detection accuracy without addressing practical deployment concerns such as scalability, multi-site management, real-time monitoring, and transparent integration with existing applications.

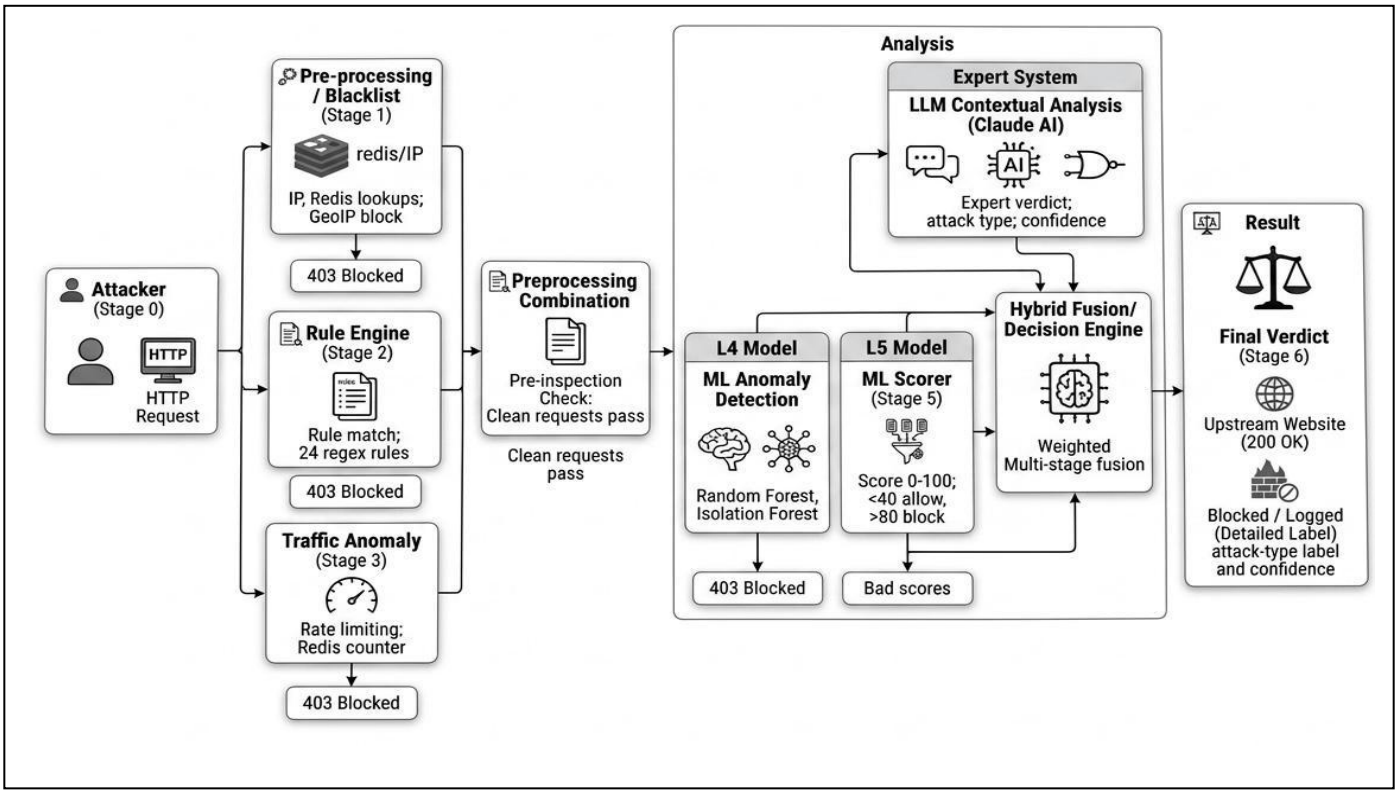
The IntelliWAF framework presented in this work addresses these challenges through a multi-layer architecture that combines

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

signature-based inspection, traffic anomaly detection, supervised machine learning using Random Forest, and unsupervised anomaly detection using Isolation Forest within a unified reverse-proxy deployment model. By integrating multiple complementary detection mechanisms, IntelliWAF aims to improve the identification of both known and unknown web attacks while maintaining deployment simplicity and real-time operational visibility.

## III. SYSTEM ARCHITECTURE

The architecture of IntelliWAF is designed around a layered reverse-proxy model that inspects incoming web traffic before it reaches the protected application. The framework combines traditional security controls with machine learning-based analysis to provide comprehensive protection against both known and unknown web threats. Figure 1 illustrates the overall architecture and request-processing workflow of the proposed system.Fig. 1. Overall IntelliWAF Architecture




The architecture consists of four major functional stages: pre-processing and traffic filtering, attack detection, machine learning analysis, and decision-making. Each stage contributes specific security information that is ultimately used to determine whether a request should be allowed, monitored, or blocked.

### A. Request Interception Layer

All incoming HTTP requests are routed through IntelliWAF before reaching the protected web application. The framework operates as a transparent reverse proxy using Nginx and mitmproxy, allowing security inspection to be performed without modifying the target application. This deployment model simplifies integration and enables protection of existing websites with minimal configuration changes.

When a client initiates a request, the reverse proxy captures the traffic and forwards it through the detection pipeline. The request is inspected layer by layer, and security decisions are applied before communication with the upstream server is established.

### B. Pre-Processing and Traffic Filtering

The first stage of the architecture performs rapid filtering operations to eliminate obviously malicious traffic with minimal computational overhead.

#### 1) IP Blacklist Layer

The blacklist module maintains a repository of blocked IP addresses using Redis. Requests originating from known malicious sources are immediately rejected with an HTTP 403

response. This mechanism reduces unnecessary processing within subsequent layers and prevents repeated attacks from previously identified sources.

#### 2) Rule-Based Detection Layer

After blacklist verification, requests are analyzed using a signature-based detection engine containing twenty-four predefined attack detection rules. The rule engine identifies common attack patterns associated with SQL Injection, Cross-Site Scripting, Path Traversal, SSRF, Command Injection, NoSQL Injection, and related web threats.

Requests matching any rule are classified as malicious and blocked immediately. This stage provides efficient detection of well-known attack techniques while maintaining low processing overhead.

#### 3) Traffic Anomaly Layer

The traffic anomaly layer performs request-rate monitoring to identify suspicious client behavior. Redis-based counters track request frequency within a configurable time window. Clients exceeding acceptable thresholds are considered potential attackers and are temporarily blocked. This layer helps mitigate brute-force attacks, automated scanning activity, and request-flooding attempts.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

**C. Request Normalization and Feature Preparation**

Requests that successfully pass the initial filtering stages undergo preprocessing before machine learning analysis. This stage prepares traffic for classification by removing encoding-related ambiguities and extracting security-relevant characteristics.

URL-encoded payloads are decoded, request content is normalized, and multiple statistical features are generated from the HTTP request. These features capture behavioral properties such as payload structure, encoded character frequency, keyword occurrences, entropy measurements, and special-character distributions.

The resulting feature vector serves as input to the machine learning components of IntelliWAF.

**D. Machine Learning Analysis Layer**

The machine learning stage consists of two complementary models designed to detect both known attack patterns and previously unseen malicious behavior.

**1) Random Forest Classification Model**

The first model utilizes Random Forest classification to distinguish legitimate requests from malicious traffic. The model is trained using HTTP request samples obtained from the CSIC 2010 dataset and evaluates extracted request features to generate a threat score.

The classifier provides high detection accuracy for attack categories represented within the training dataset and enhances detection capabilities beyond traditional signature-based approaches.

**2) Isolation Forest Anomaly Detection Model**

To improve resilience against emerging threats, IntelliWAF incorporates an Isolation Forest model for anomaly detection. Unlike supervised classifiers, Isolation Forest identifies requests that deviate significantly from normal traffic behavior.

This layer enables the detection of suspicious requests that may not match predefined signatures or previously observed attack patterns, thereby improving protection against zero-day and obfuscated attacks.

**E. Hybrid Decision Engine**

The outputs produced by the rule engine, Random Forest classifier, and Isolation Forest detector are consolidated within a hybrid decision engine. The decision engine evaluates evidence collected from multiple detection layers and calculates an overall threat assessment for each request.

By combining deterministic rule matching with machine learning predictions and anomaly scores, the framework reduces dependence on any single detection mechanism and improves overall detection reliability.

Based on the final evaluation, the decision engine performs one of the following actions:

- Forward the request to the protected application.
- Record the request for monitoring and analysis.
- Block the request and generate a security event.

**I. F. Logging and Monitoring Infrastructure**

All security events generated by IntelliWAF are transmitted to a Redis-backed event queue. Separating traffic processing from logging operations prevents database interactions from affecting request-inspection performance.

A background logging service retrieves events from the queue and stores them within a MySQL database. The collected information is subsequently displayed through a monitoring dashboard that provides administrators with real-time visibility into attack activity, blocked requests, detection statistics, and system status.

**G. Architectural Advantages**

The proposed architecture offers several advantages over conventional WAF deployments:

- Multi-layer defense against both known and unknown attacks.
- Transparent deployment through reverse-proxy integration.
- Support for machine learning-assisted threat detection.
- Real-time attack monitoring and event logging.
- Scalable Docker-based deployment architecture.
- Modular design that supports future enhancements and additional detection mechanisms.

The combination of rule-based inspection, anomaly detection, and machine learning analysis enables IntelliWAF to provide a balanced approach to web application security while maintaining practical deployability in real-world environments.

## IV. METHODOLOGY

The IntelliWAF framework employs a six-layer security architecture that combines signature-based detection, traffic anomaly monitoring, supervised machine learning, and unsupervised anomaly detection to mitigate web application attacks in real time. The proposed methodology is designed to analyze incoming HTTP requests at multiple levels before determining whether a request should be allowed, monitored, or blocked.

The overall detection workflow consists of request interception, preprocessing, attack signature inspection, machine learning analysis, anomaly detection, hybrid threat scoring, and final decision generation. By integrating multiple complementary detection mechanisms, the framework reduces dependence on a single security technique and improves resilience against both known and previously unseen attack patterns.

- **A. Request Acquisition and Preprocessing**

All incoming HTTP requests are intercepted through the reverse-proxy infrastructure before reaching the protected web application. During preprocessing, request parameters, URLs, headers, and payload content are collected and normalized.

To prevent evasion through encoding techniques, URL decoding and payload normalization are performed before inspection. The normalized request is subsequently forwarded to the detection pipeline for security analysis.

Let an incoming request be represented as:

$$R=\{URL,Headers,Payload,IP\}$$

where:

- (URL) represents the requested resource.
- (Headers) represent HTTP header information.
- (Payload) represents request content.
- (IP) represents the client source address.

- **B. Layer 1 – IP Reputation Filtering**

The first layer performs blacklist verification using stored malicious IP addresses. Requests originating from known malicious sources are immediately rejected before further processing.

The blacklist function can be expressed as:

$$B(IP)=\begin{cases}1, & \text{if IP exists in blacklist}\\ 0,&\text{otherwise}\end{cases}$$

where (B(IP)=1) indicates a blocked source.

- **C. Layer 2 – Rate Limiting and Traffic Monitoring**

The second layer analyzes request frequency to identify brute-force attacks, automated scanning, and denial-of-service attempts.

For a given client IP:

$$RR=\frac{N_r}{T}$$

where:

- (RR) = Request Rate
- (N_r) = Number of requests

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

- (T) = Observation time window

If the request rate exceeds the predefined threshold (RR_{max}), the request is classified as suspicious and may be blocked.

- **D. Layer 3 – Rule-Based Attack Detection**

The third layer employs a set of predefined security rules implemented using regular expressions. The rule engine detects common web attacks including:

- SQL Injection
- Cross-Site Scripting (XSS)
- Path Traversal
- SSRF
- Command Injection
- NoSQL Injection

The rule evaluation function is defined as:

$$\text{Rule}(R)=\begin{cases}1, & \text{if attack signature detected}\\ 0, & \text{otherwise}\end{cases}$$

A value of 1 indicates that the request matches a known attack pattern.

- **E. Layer 4 – Feature Extraction and Random Forest Classification**

Requests that pass signature inspection undergo feature extraction. The extracted features include:

- Payload length
- Character entropy
- Special-character ratio
- Encoded sequence frequency
- Keyword occurrence count
- URL complexity metrics

The resulting feature vector is represented as:

$$F=\{f_1,f_2,f_3,\ldots,f_n\}$$

where $(f_i)$ represents an individual security feature.
The feature vector is then evaluated using a Random Forest classifier trained on HTTP traffic samples derived from the CSIC 2010 dataset. The classifier produces a maliciousness probability:

$$RF(R)=P(\text{Malicious}|F)$$

where:

$$0 \le RF(R) \le 1$$

Higher values indicate a greater likelihood of malicious activity.

- **F. Layer 5 – Isolation Forest Anomaly Detection**

Although supervised learning models perform well on known attack patterns, they may struggle against previously unseen attacks. To address this limitation, IntelliWAF employs Isolation Forest anomaly detection.
The anomaly score generated by Isolation Forest is represented as:

$$IF(R)=A_s$$

where:

- $(A_s)$ = Anomaly Score

Higher anomaly scores indicate stronger deviation from normal traffic behavior.
This layer assists in identifying zero-day attacks and uncommon attack variants that may not appear in training datasets.

- **G. Layer 6 – Hybrid Threat Assessment**

The outputs generated by the rule engine, Random Forest classifier, and Isolation Forest detector are combined within a hybrid decision engine.
The overall threat score is computed as:

$$TS=\alpha\ Rule(R)+\beta\ RF(R)+\gamma\ IF(R)$$

where:

- (TS) = Threat Score
- (Rule(R)) = Rule-based detection output
- (RF(R)) = Random Forest score
- (IF(R)) = Isolation Forest anomaly score
- $(\alpha,\beta,\gamma)$ = weighting coefficients

subject to:

$$\alpha+\beta+\gamma=1$$

The weighting coefficients determine the contribution of each detection mechanism toward the final security decision.

- **H. Final Decision Generation**

The final verdict is generated using threshold-based classification.

$$\text{Decision}(R)=\begin{cases}\text{Block}, & TS \ge T_b\\ \text{Monitor}, & T_m \le TS < T_b\\ \text{Allow}, & TS < T_m\end{cases}$$

where:

- $(T_b)$ = Blocking Threshold
- $(T_m)$ = Monitoring Threshold

Requests classified as **Block** are rejected before reaching the protected application. Requests classified as **Monitor** are logged for further analysis, while **Allow** requests are forwarded to the protected web application.

- **I. Event Logging and Monitoring**

All security events generated throughout the detection pipeline are transmitted to a Redis-backed event queue. The queue decouples traffic inspection from database operations, ensuring efficient request processing. Logged events are subsequently stored in MySQL and visualized through the monitoring dashboard for real-time security analysis.
The proposed methodology enables IntelliWAF to combine deterministic attack detection with machine learning and anomaly detection techniques, thereby improving its ability to identify both known and previously unseen web application threats.

## V. RESULTS

This section evaluates the effectiveness of the proposed IntelliWAF framework in detecting web application attacks and maintaining secure traffic inspection. The evaluation focuses on attack detection capability, machine learning performance, system functionality, and real-time monitoring within the multi-layer architecture.

### A. Research Hypothesis

To assess the effectiveness of the proposed framework, the following hypotheses were considered:

**Null Hypothesis ($H_0$):**
The IntelliWAF framework does not provide a meaningful improvement in web attack detection compared to conventional signature-based Web Application Firewall solutions.

**Alternative Hypothesis ($H_1$):**
The IntelliWAF framework improves web attack detection through

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

the integration of signature-based inspection, Random Forest classification, and Isolation Forest anomaly detection within a multi-layer security architecture.

The experimental evaluation presented in this section examines the framework's ability to identify multiple categories of web attacks while maintaining operational functionality and real-time monitoring capabilities.

**B. Experimental Environment**

The IntelliWAF framework was implemented using Python, Nginx, mitmproxy, Redis, MySQL, Docker, and Scikit-learn. The system was deployed as a reverse-proxy Web Application Firewall where all incoming requests were inspected before reaching the protected web application.

The Random Forest classifier was trained using HTTP traffic samples derived from the CSIC 2010 dataset [2], while the Isolation Forest model was integrated to identify anomalous request behavior. Security events generated during request inspection were processed through a Redis-backed event queue and displayed through a real-time monitoring dashboard.

**C. Attack Detection Evaluation**

The detection capability of IntelliWAF was evaluated against several common web application attack categories. The rule engine, machine learning classifier, and anomaly detection layer collectively analyzed incoming requests before generating a final security decision.

**Table I:**
**Detection Results for Common Web Attacks**

| Attack Category | Detection Status |
|---|---|
| SQL Injection | Detected |
| Cross-Site Scripting (XSS) | Detected |
| Path Traversal | Detected |
| SSRF | Detected |
| Command Injection | Detected |
| NoSQL Injection | Detected |
| Log4Shell Pattern Detection | Detected |

The results indicate that IntelliWAF successfully identified malicious requests associated with all tested attack categories. Signature-based detection effectively blocked known attack patterns, while machine learning analysis improved detection of encoded and modified payloads that may not directly match predefined signatures.

**D. Machine Learning Evaluation**

The machine learning component of IntelliWAF consists of a Random Forest classifier and an Isolation Forest anomaly detector.

The Random Forest model achieved an accuracy of approximately **91.5%** when evaluated using HTTP traffic samples derived from the CSIC 2010 dataset. This result demonstrates the capability of ensemble learning techniques to distinguish malicious requests from legitimate web traffic.

The Isolation Forest model complemented the classifier by identifying abnormal traffic behavior that deviated from learned request patterns. The combination of supervised classification and anomaly detection improved the framework's ability to identify suspicious requests that may bypass traditional signature-based inspection.

**E. Functional Evaluation**

In addition to attack detection, the operational functionality of the framework was evaluated to verify successful integration of all architectural components.

**Table II:**
**System Functionality Evaluation**

| Functionality | Result |
|---|---|
| Reverse Proxy Operation | Successful |
| Multi-Site Routing | Successful |
| Redis Event Queue Processing | Successful |
| MySQL Log Storage | Successful |
| Dashboard Integration | Successful |
| Dynamic Website Onboarding | Successful |
| Attack Logging | Successful |
| Real-Time Monitoring | Successful |

The evaluation confirmed that all major system components operated correctly within the proposed architecture. Incoming requests were successfully routed through the detection pipeline, while security events were recorded and displayed through the monitoring dashboard.

**F. Real-Time Monitoring and Logging**

The Redis-backed event queue successfully decoupled traffic inspection from database operations. Security events generated by the detection pipeline were processed independently and stored within the logging infrastructure without interrupting request analysis.

This architecture enabled continuous monitoring of attack activity while maintaining efficient request processing. Administrators were able to observe blocked attacks, request statistics, and security events through the dashboard in near real time.

**G. Discussion**

The evaluation results support the alternative hypothesis ($H_1$) by demonstrating that a multi-layer security architecture provides broader detection coverage than a traditional signature-only approach. While the rule engine rapidly identified known attack signatures, the Random Forest and Isolation Forest models contributed additional analytical capabilities for identifying suspicious and anomalous requests.

The results further demonstrate that IntelliWAF successfully combines reverse-proxy deployment, machine learning-based threat detection, anomaly analysis, event-driven logging, and real-time monitoring within a single security platform. The framework therefore provides an effective and deployable solution for protecting web applications against a diverse range of web-based threats.

## VI. LIMITATIONS AND FUTURE WORK USING THE TEMPLATE

Although IntelliWAF demonstrates effective detection of multiple web application attack categories through its multi-layer security architecture, several limitations remain that can be addressed in future versions of the framework.

**A. Limitations**

**1) Dependence on Training Data**

The effectiveness of the Random Forest classifier depends on the quality and diversity of the training dataset. In the current implementation, the model is trained using the CSIC 2010 HTTP dataset. While this dataset contains a variety of legitimate and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

malicious HTTP requests, modern attack techniques continue to evolve and may exhibit characteristics that differ from those represented in the training data. Consequently, the classification model may require periodic retraining using more recent datasets to maintain optimal detection performance.

**2) Limited Detection of Highly Sophisticated Attacks**

The signature-based detection layer effectively identifies known attack patterns such as SQL Injection, Cross-Site Scripting, SSRF, and Path Traversal attacks. However, highly sophisticated attacks that employ advanced obfuscation techniques, multi-stage payloads, or application-specific exploitation methods may partially evade traditional rule-based inspection mechanisms.

**3) WebSocket Traffic Inspection**

The current implementation primarily focuses on HTTP request inspection. While WebSocket upgrade requests can be monitored, deep inspection of WebSocket message payloads is not fully implemented. As modern web applications increasingly rely on real-time communication channels, comprehensive WebSocket traffic analysis remains an area for future enhancement.

**4) Infrastructure Dependency**

The framework relies on supporting services such as Redis and MySQL for event processing, logging, and configuration management. Service interruptions affecting these components may impact monitoring functionality and reduce operational visibility, although core request inspection remains available through the reverse-proxy architecture.

**5) Resource Consumption**

The integration of multiple detection layers improves security coverage but introduces additional computational overhead compared to traditional signature-only WAF solutions. As traffic volume increases, efficient resource management and horizontal scalability become important considerations for large-scale deployments.

**B. Future Work**

Several enhancements can further improve the effectiveness, scalability, and intelligence of the IntelliWAF framework.

**1) Advanced Threat Intelligence Integration**

Future versions of IntelliWAF can incorporate external threat intelligence feeds to automatically identify malicious IP addresses, emerging attack campaigns, and newly discovered vulnerabilities. Such integration would enable more proactive threat mitigation and improve blacklist management.

**2) AI-Assisted Security Analysis**

A promising extension involves the integration of Large Language Models (LLMs) or AI-assisted reasoning systems to analyze suspicious requests that cannot be confidently classified by existing detection layers. Such systems could provide contextual explanations, attack categorization, and intelligent security recommendations for administrators. This functionality is proposed as a future enhancement and is not part of the current implementation.

**3) Enhanced Machine Learning Models**

Future research may explore advanced machine learning and deep learning techniques, including neural networks, ensemble anomaly detection frameworks, and adaptive learning models. These approaches may improve detection performance against highly obfuscated and previously unseen attack patterns.

**4) WebSocket and API Security Inspection**

Extending the framework to support deep inspection of WebSocket traffic, REST APIs, GraphQL endpoints, and modern web service architectures would broaden its applicability to contemporary web applications.

**5) Cloud and Container-Oriented Deployment**

Future development may focus on large-scale deployment through cloud-native technologies such as Kubernetes and container orchestration platforms. Such enhancements would improve scalability, fault tolerance, and operational flexibility for enterprise environments.

**6) Security Information and Event Management (SIEM) Integration**

Integration with SIEM platforms would enable centralized security monitoring, correlation of attack events, and automated incident response. This capability would improve situational awareness and strengthen enterprise security operations.

**C. Summary**

The current IntelliWAF implementation demonstrates the practicality of combining rule-based inspection, machine learning classification, and anomaly detection within a unified Web Application Firewall architecture. Addressing the identified limitations and implementing the proposed future enhancements can further improve detection accuracy, scalability, and adaptability against emerging web threats, thereby strengthening the overall effectiveness of the framework.

## VII. CONCLUSION

This paper presented **IntelliWAF**, a six-layer intelligent Web Application Firewall designed to improve the detection and mitigation of modern web-based threats. Traditional WAF solutions primarily depend on static signature databases, which are effective against known attacks but often struggle to identify obfuscated payloads, evolving attack techniques, and previously unseen malicious behaviors. To address these challenges, IntelliWAF combines multiple security mechanisms, including IP blacklist filtering, rate limiting, signature-based detection, Random Forest classification, and Isolation Forest anomaly detection within a unified reverse-proxy architecture.

The proposed framework was implemented using Nginx, mitmproxy, Redis, MySQL, Docker, and Scikit-learn, enabling transparent deployment without requiring modifications to the protected web application. By integrating rule-based inspection with machine learning and anomaly detection techniques, IntelliWAF provides multiple layers of analysis that collectively improve threat detection coverage. The system successfully demonstrated the detection of common web application attacks, including SQL Injection, Cross-Site Scripting (XSS), Server-Side Request Forgery (SSRF), Path Traversal, Command Injection, and related malicious request patterns. In addition, the Redis-backed event pipeline and monitoring dashboard enabled real-time visibility into security events while maintaining efficient request processing.

The evaluation results indicate that the proposed multi-layer architecture offers a practical balance between detection capability and deployment simplicity. The Random Forest classifier achieved an accuracy of approximately 91.5% on HTTP traffic samples derived from the CSIC 2010 dataset, while the Isolation Forest model provided additional support for identifying anomalous and potentially unknown attack behaviors. These findings support the use of hybrid security architectures that combine deterministic detection techniques with data-driven analysis methods.

In conclusion, IntelliWAF demonstrates that integrating signature-based inspection, machine learning classification, and anomaly detection within a layered security framework can significantly enhance web application protection. The proposed architecture provides a scalable foundation for future improvements, including advanced threat intelligence integration, AI-assisted security analysis, cloud-native deployment, and enhanced monitoring capabilities. As web applications continue to evolve, frameworks such as IntelliWAF can contribute to the development of more adaptive and resilient web security solutions.

## VIII. REFERENCES

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

[1] OWASP Foundation, "OWASP Top Ten 2021," Open Web Application Security Project, 2021. [Online]. Available: https://owasp.org/www-project-top-ten/

[2] D. Giménez, G. Villegas, and G. Maciá-Fernández, "HTTP Dataset CSIC 2010," Spanish National Research Council (CSIC), 2010. [Online]. Available: https://www.impactcybertrust.org/dataset_view?idDataset=940

[3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM)*, Pisa, Italy, Dec. 2008, pp. 413–422. Available: https://ieeexplore.ieee.org/document/4781136

[4] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. Available: https://link.springer.com/article/10.1023/A:1010933404324

[5] J. Zhang, M. Zulkernine, and A. Haque, "Random-Forests-Based Network Intrusion Detection Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 38, no. 5, pp. 649–659, Sep. 2008. Available: https://ieeexplore.ieee.org/document/4629028

[6] S. Waskle, L. Parashar, and U. Singh, "Intrusion Detection System Using PCA with Random Forest Approach," in *Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, 2020, pp. 803–808.

[7] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying Convolutional Neural Network for Network Intrusion Detection," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Udupi, India, 2017, pp. 1222–1228.

[8] C. Torrano-Gimenez, A. Perez-Villegas, and G. Alvarez, "An Anomaly-Based Web Application Firewall Using HTTP-Specific Features and One-Class Classifiers," in *Proceedings of the International Conference on Computational Intelligence and Security (CIS)*, Beijing, China, 2010, pp. 23–28.

[9] M. Amouei, M. Rezvani, and M. Fateh, "RAT: Reinforcement-Learning-Driven and Adaptive Testing for Vulnerability Discovery in Web Application Firewalls," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3371–3386, Sep./Oct. 2022.

[10] D. Mitropoulos, P. Louridas, M. Polychronakis, and A. D. Keromytis, "Defending Against Web Application Attacks: Approaches, Challenges and Implications," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 2, pp. 188–203, Mar./Apr. 2019. Available: https://ieeexplore.ieee.org/document/8248612

[11] A. Azer and M. Sobh, "Evaluating the ModSecurity Web Application Firewall Against SQL Injection Attacks," in *Proceedings of the 15th International Computer Engineering Conference (ICCES)*, Cairo, Egypt, 2020, pp. 1–6.

[12] S. Bhatt, P. K. Manadhata, and L. Zomlot, "The Operational Role of Security Information and Event Management Systems," *IEEE Security & Privacy*, vol. 12, no. 5, pp. 35–41, Sep./Oct. 2014.

[13] A. Applebaum, T. Gaber, and A. Ahmed, "Signature-Based and Machine-Learning-Based Web Application Firewalls: A Short Survey," *Procedia Computer Science*, vol. 189, pp. 359–367, 2021.

[14] T. Mahoney, K. Bhatia, and M. Natu, "On the Performance of Isolation Forest and Multi Layer Perceptron for Anomaly Detection in Industrial Control System Networks," in *Proceedings of the IEEE Conference on Communications and Network Security (CNS)*, 2022, pp. 1–6.

[15] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL Injection Based on Artificial Neural Network," *Knowledge-Based Systems*, vol. 190, p. 105528, Feb. 2020.

[16] A. Sharma, T. Pal, M. Bose, and S. Singh, "Detection of SQL Injection and XSS Attacks in Three-Tier Web Applications," in *Proceedings of the International Conference on Information Technology (InCITe)*, Noida, India, 2017, pp. 1–6.

[17] B. Anam, S. Noor, and H. Ahmad, "Security Audit of Docker Container Images in Cloud Microservices," in *Proceedings of the 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC)*, Jalandhar, India, 2021, pp. 1–6.

[18] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. Available: https://jmlr.org/papers/v12/pedregosa11a.html

[19] mitmproxy Project Contributors, "mitmproxy: Interactive HTTPS Proxy," [Online]. Available: https://mitmproxy.org/

[20] R. Sahoo, A. Lath, and A. Choudhary, "Leveraging Docker Containers for Deployment of Web Applications in Microservices Architecture," in *Proceedings of the IEEE International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, Dubai, UAE, 2024, pp. 1–6.

[21] Docker Inc., "Docker Documentation," [Online]. Available: https://www.docker.com/

[22] Nginx Inc., "NGINX Documentation," [Online]. Available: https://nginx.org/en/docs/

[23] Redis Ltd., "Redis Documentation," [Online]. Available: https://redis.io/docs/

[24] Oracle Corporation, "MySQL Documentation," [Online]. Available: https://dev.mysql.com/doc/