ML-Enhanced Behavioral & Anomaly Detection Web Application Firewall

Mr.M.Asan Nainar[1], B.K Sibikarthik[2]

[1]*Assistant professor, Department of Computer Applications, SRM Valliammai Engineering College, Anna University, Chennai.*

[2]*PG Student, Department of Computer Applications, SRM Valliammai College, Anna University, Chennai.*

***Abstract*—Traditional firewalls are frequently unable to identify complex user-based threats in the constantly changing field of web security. A Machine Learning-Enhanced Behavioral and Anomaly Detection Web Application Firewall (WAF) is presented in this research. It actively tracks, examines, and reacts to unusual user activity. Two web applications are used to implement the system: a secure application with anomaly detection features and another that is purposefully left open for passive observation. A real-time dashboard is available in both applications to track user entry and exit times as well as behavioral trends.**

Comprehensive behavior tracking is made possible via MongoDB, which acts as the backend for gathering and storing activity records. By examining input payloads and user interaction sequences, the secure application uses a trained machine learning model that can recognize different attack patterns, including brute force, XSS, and SQL injection. Potential breaches are effectively avoided since the system immediately bans access and reroutes the user to the login page when it detects suspicious activity.

This research shows how to combine machine learning and behavioral analytics to improve web application defenses, providing a dynamic and flexible security solution that goes well beyond static rule-based solutions.

1. INTRODUCTION

Although web applications are a crucial component of contemporary digital infrastructure, they are becoming more and more susceptible to various cyberattacks, including brute-force attacks, SQL injection, and cross-site scripting (XSS). Static rule sets and signature-based detection are key components of traditional Web application firewalls (WAFs), but they are sometimes insufficient to stop new and changing attack patterns. By combining behavioral analysis and machine learning (ML) in a WAF system, this project suggests an intelligent and flexible security solution to these constraints.This project's objective is to create and deploy an ML-Enhanced Behavioral and Anomaly Detection Web Application Firewall that actively reacts to questionable activity in real time in addition to monitoring user activity. There are two applications in the system

1. A web application that is insecure and used to observe user behavior without any limitations.

2. A safe online program that recognizes and stops unusual activity using machine learning

.

The activity logs, which are maintained using MongoDB for analysis, and user entrance and exit times are tracked via a dashboard in both systems. Based on user input and interaction patterns, the machine learning model may identify and forecast possible future attacks after being trained on a dataset of known attack patterns.This intelligent firewall provides a strong solution for contemporary web applications by improving security and responding to emerging threats. The initiative emphasizes how important it is to integrate behavioral analysis and machine learning to build more intelligent and robust cybersecurity solutions.

LITERATURE REVIEW

In order to identify malicious activity, traditional Web Application Firewalls (WAFs) mostly rely on rule-based techniques and signature matching. These techniques work well for recognized threats, but they frequently miss behavioral irregularities, disguised inputs, and zero-day assaults. In order to improve detection skills, researchers have thus investigated the combination of machine learning (ML) and user behavior analytics (UBA).

Early sentiment and pattern analysis methods were presented by Pang et al. (2002), proving that machine learning models are capable of successfully spotting minute patterns in unstructured data. Later behavioral analysis in cybersecurity was made possible by this. In a similar vein, Sommer and Paxson (2010) pointed out that strict pattern definitions in conventional anomaly detection systems result in significant false positive rates. They underlined the necessity of adaptable and context-aware models.

By training on payload datasets, recent studies like Sharma et al. (2021) suggested using supervised learning models to identify SQL injection and XSS assaults. Although their method achieved great accuracy, it needed to be updated with new datasets on a regular basis to be successful. This was extended by Tang et al. (2022) by adding deep learning methods for real-time user behavior monitoring. By examining trends in user navigation, time spent, and input sequences, their system was able to successfully lower false positives.

MongoDB and other NoSQL databases have been widely used for real-time logging and quick querying of behavioral data due to its scalability and storage capabilities. This helps ML-driven security systems identify and react quickly.

The research currently in publication, which emphasizes the efficiency of behavior-based detection systems in spotting complex threats, generally supports the integration of machine learning (ML) with online security solutions. The majority of research, however, indicates that the most secure and flexible approach is achieved by integrating many strategies, including activity tracking, payload analysis, and machine learning prediction.

2. SYSTEM OVERVIEW

2.1 EXISTING SYSTEM

2.1.1 A SPECTRUM OF ERROR TYPES:

1. False Positives (Type I Errors):
Type I errors, often known as false positives, happen when benign user activity is mistakenly reported as harmful. This may result in a bad user experience, needless account bans, and a decline in system confidence.

2. False Negatives (Type II Errors):
When real harmful activity remains unnoticed, attackers can get around security safeguards and take advantage of program flaws, possibly leading to serious data breaches. This is known as a false negative

3. Misclassification of Abnormal Behavior:
When trained on sparse or imbalanced datasets, machine learning models may find it difficult to discern between uncommon but acceptable behaviors and real dangers, which lowers detection accuracy.

2.1.2 DRAWBACKS OF THE EXISTING SYSTEM:

1. Static Rule-Based Detection:
Because conventional WAFs rely on preset rules and signatures, they are unable to fend off developing intrusion tactics, zero-day assaults, and polymorphic threats.

2. High False Alarm Rate:
Current systems frequently produce a disproportionate number of false positives, classifying benign user activity as malicious, which impairs user experience and adds to administrative workload.

3. Lack of Real-Time Adaptive Response:
The majority of systems in use today are unable to instantly adjust to new attack vectors. Because of their inability to learn, human upgrades are necessary to keep security efficacious.

2.2 NEED FOR NEW SYSTEM:

1. Increase in Complex Behavioral Attacks: To get over conventional security measures, contemporary attackers frequently imitate typical user behavior. Behavior analysis must be included into a new system in order to identify insider threats and minor irregularities.

2. Using Machine Learning to Make Wise Decisions:

The firewall can continually evolve and adjust to new threats by integrating machine learning (ML), which allows the system to learn from past data and make intelligent predictions.

3. Thorough Monitoring and Data Gathering:
Proactive protection and post-attack analysis are made possible by storing user entry time, exit time, and activity logs in a NoSQL database such as MongoDB. This provides profound insights into user behavior.

2.3 PROPOSED SYSTEM:
The suggested solution offers sophisticated defense against contemporary online threats by introducing an intelligent, adaptive Web Application Firewall (WAF) that combines User Behavior Analytics (UBA) and Machine Learning (ML). Its two primary components are a safe online application that actively identifies and stops suspicious activity in real time, and a vulnerable web application that monitors user behaviors without interfering.

The following are important aspects of the suggested system:

1. ML-Based Anomaly Detection:
Based on user input and behavioral anomalies, a machine learning model is trained using historical attack patterns (such as script injection, SQL injection, and brute-force inputs) to detect malicious activity.

2. Behavior Monitoring and Logging:
All activity logs are stored in MongoDB for analysis and training, and the system keeps track of comprehensive user behavior, including login/logout timings, session lengths, page browsing patterns, and input sequences.

3. Real-Time Threat Response:
To stop more exploitation or data leakage, the system instantly takes action upon detecting an abnormality. This response may include barring the user or rerouting them to the login page.

4. Dashboard for Admin Insight:
To assist ongoing monitoring and threat analysis, an integrated dashboard offers real-time insights into user actions, threats identified, and application use data.

2.3.1 FEATURES

Machine Learning-Based Threat Detection:
Machine learning-based threat detection makes use of trained machine learning models to identify and categorize harmful inputs, including script-based assaults or SQL injection, enabling smart threat prediction and avoidance.

User Behavior Analytics (UBA):
Tracks user activity continuously, including input behavior, navigation patterns, and entry/exit times, to identify any irregularities that could point to abuse or illegal access.

Real-time dashboard monitoring:
An integrated dashboard shows activity logs, security warnings, and user behavior in real-time, assisting administrators in seeing any dangers and taking prompt, effective action.

3 SYSTEM ANALYSIS

Proposed System Analysis:
To identify and stop anomalous activity, the proposed ML-enhanced WAF uses machine learning to monitor user behavior in real time. Through automated blocking and intelligent decision-making, it increases the accuracy of threat detection, adjusts to new attack methods, and guarantees improved user session control.

3.1 FEASIBILITY STUDY

3.1.1 ECONOMICAL FEASIBILITY

Cost-effective Development with Open-Source Tools: Python, Flask, and MongoDB are just a few of the open-source technologies used in the system's construction, which drastically lowers development and deployment expenses without sacrificing flexibility or performance.

Decreased Long-Term Operational Costs: The system's integration of automation and machine learning lessens the need for ongoing human oversight and manual rule modifications, which eventually results in cheaper maintenance and operating costs.

3.1.2 TECHNICAL FEASIBILITY

Integration of Reliable Technologies:
The system makes use of dependable technologies that are resilient, scalable, and extensively maintained, including Flask for web frameworks, MongoDB for database operations, and Python for backend functionality.

Machine Learning Compatibility:
The architecture facilitates easy model training, testing, and real-time anomaly detection prediction by integrating with well-known ML libraries such as Scikit-learn or TensorFlow.

Real-Time Monitoring and Response: Without the need for human interaction, the system can technically manage real-time data collecting, analysis, and automatic actions like barring users or setting off alarms, guaranteeing prompt threat mitigation.

3.1.3 SOCIAL FEASIBILITY

Enhanced User Safety and Trust:
The system fosters a safe environment by employing intelligent behavior analysis to defend online applications against cyber attacks. This raises user confidence and trust in digital platforms.

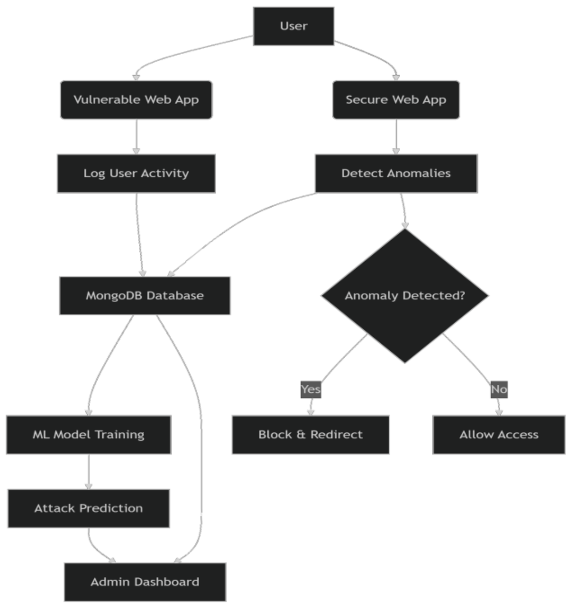
Cybersecurity Education and Awareness:
By promoting knowledge of contemporary attack methods and emphasizing the value of anomaly detection, the implementation helps create a society that is more cognizant of cyber threats.

Positive Effect on Businesses and Users: Businesses can reassure their customers and users of robust security standards, which enhances customer happiness, builds brand recognition, and makes the internet a safer place for all users.

Ethical Technology Use:
By identifying dangers and encouraging the proper deployment of AI and machine learning in security applications, the system guarantees user privacy is preserved.

Enhanced Awareness of Security:
By automatically identifying and stopping unusual activity, the initiative raises user awareness of security. Better user behavior overall and a deeper comprehension of online safety procedures might result from this.
.

1.0 Web application firewall flow diagram



3.2 TABLE DESIGN

Monitors all user interactions, security actions, and anomaly detections based on machine learning across your secure and insecure web apps.

1.2 Table Design

sno	Field	Type	Description
1	id	String	Unique log ID (UUID or ObjectId)
2	user_id	String	User ID or "anonymous"
3	app_type	String	"vulnerable" or "secure"
4	timestamp	DateTime	Activity time
5	action	String	What user did (e.g., "login", "search")
6	details	Text	Extra info (simplified JSON)

7	is_attack	Boolean	true if ML detected an attack
8	blocked	Boolean	true if action was blocked

Table Description

1. event_id (String): Unique identifier for each logged event (e.g., UUID).
2. user_ref (String): Anonymous or authenticated user identifier (e.g., "user-123" or "anon-192.168.1.1").
3. app_type (String): Either "vulnerable" (no blocking) or "secure" (WAF-protected).
4. timestamp (ISODate): Exact time of the user action.
5. activity (String): Type of activity (e.g., "login", "form_submit", "file_upload").
6. request (JSON): Sanitized request details (URL, headers, payload snippets).
7. ml_analysis (JSON): Contains:
8. verdict (String: "normal", "suspicious", "malicious").
9. risk_score (Float: 0.0–1.0, confidence level).
10. threat_type (String, optional: "SQLi", "XSS", etc.).
11. action (String): System response ("allowed", "blocked", "redirected").

3.3 REPORT GENERATION

An effective and automatic report creation tool in the system offers comprehensive insights into user behavior and security incidents. These reports are essential for tracking, evaluating, and enhancing the Web Application Firewall's functionality.

Important elements in creating reports include:
Administrators may better understand how users engage with the program by using User Activity Reports, which record session length, entrance and exit timings, and user navigation activity.

Anomaly discovery Logs:
Provides comprehensive documentation of all threats found, including the moment of discovery, the anomaly kind (such as script attack or SQL injection), and the system response (such as block or redirect).

ML Prediction Summary:
The machine learning model's predictions, including accuracy, false positives found, and learning performance over time, are summarized in the ML Prediction Summary.

Graphical Dashboard Reports:
Real-time summaries of assaults, user patterns, and system reactions are presented in visual charts and tables, which improve threat assessment and decision-making.

Exportable forms:
For future use, compliance, and interaction with external security products, reports may be exported in CSV or PDF forms.

4. SYSTEM IMPLEMENTATION

All of the modules needed to construct the ML-Enhanced Behavioral & Anomaly Detection Web Application Firewall are actually developed and deployed during the system installation phase. Every component is made to carry out a particular function that supports data logging, machine learning-based intelligent decision-making, and overall system security.

4.1 MODULE DESCRIPTION

The primary modules of the suggested system are as follows:

1. Vulnerable Web Application Module:
A deliberately unsafe online application created to mimic common security issues is known as a vulnerable web application module. It acts as training data for the machine learning model and records every user action to assist in identifying possible attack behaviors.

2. Secure Web Application Module:
An application with the ML-enhanced WAF that is secure. By rerouting the user to the login page, it automatically bans suspicious activity, detects abnormalities, and records user behavior in real-time.

3.User Behavior Tracking Module:
This module records the duration of the session, the time the user enters and exits, and the flow of

navigation. Every action is kept in MongoDB for auditing and analysis. It makes it possible to see user behavior in great detail.

4. Machine Learning Detection Module: This module categorizes user inputs (such as normal versus attack) by training and applying ML models. Based on patterns of learning behavior, it forecasts possible risks and initiates automatic responses to anomalous activity.

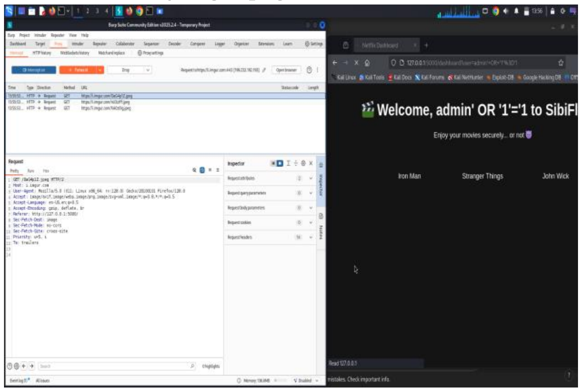
5. Dashboard and Reporting Module: Provides a real-time overview of system activities, including alarms, user logs, and the outcomes of machine learning predictions. It enables administrators to keep an eye on persistent behavior and produce reports for additional analysis.

SCREENSHOTS:

The main screenshots taken throughout the creation and implementation of the ML-Enhanced Behavioral & Anomaly Detection Web Application Firewall are shown in this part. These screenshots provide visual evidence of the user interface and capabilities that have been developed.

1. Vulnerable Web Application – Home Page
insecure Web program-Home Page Description: Shows the fundamental user interface of the program that is insecure and used to track user activity without any security controls.

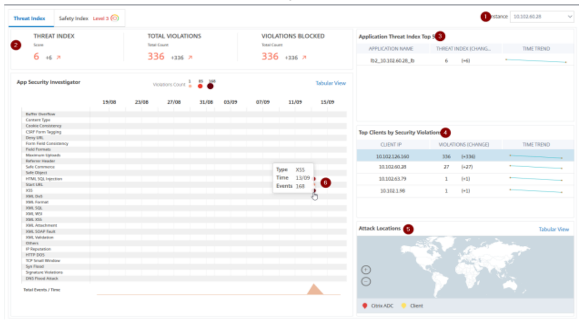
1.3 vulnerability login page



Welcome, admin' OR '1'='1 to SibiFl
Enjoy your movies securely... or not
Iron Man
Stranger Things
John Wick

2.Secure Web Application – Dashboard Dashboard, a Secure Web Application Description: Real-time user tracking (session time, login/logout) is shown on a secure application dashboard.

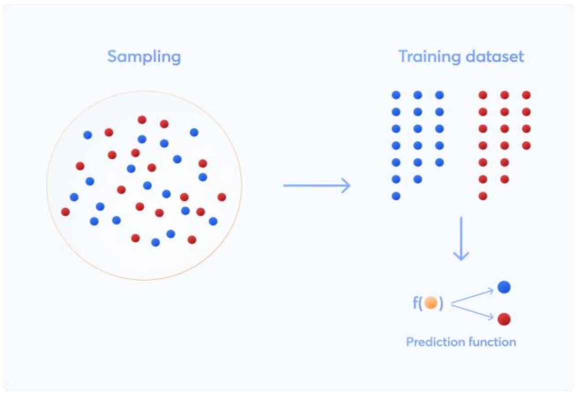
1.4 Dashboard of user entry and exit



3. MongoDB-User Activity Data
 Description: An example of data illustrating how activities, behavior logs, and user session times are kept in MongoDB.

4. ML Model Prediction Result
Description: The output that displays the ML model's prediction result, such as "Normal Activity" or "Potential Attack."

1.5 Machine learning training model



5. Auto Redirect After Attack:
This screenshot illustrates how, upon identifying anomalous activity, the user is automatically redirected to the login page.

5. CONCLUSION

The successful incorporation of machine learning into contemporary online security systems is demonstrated by the suggested ML-Enhanced Behavioral & Anomaly Detection online Application Firewall.

Through real-time anomaly detection and analysis of user behavior patterns, the technology not only improves web application security but also adjusts to changing attack tactics. Effective training and testing of the ML model is made possible by the deployment of a secure application for threat mitigation and a susceptible application for behavior observation. User activity logs are efficiently stored and retrieved for additional analysis and reporting thanks to MongoDB's integration.This system demonstrates how danger identification and response may be automated through the practical use of artificial intelligence, greatly lowering the need for manual monitoring. In order to increase detection accuracy, the model may be refined in subsequent iterations by integrating deep learning techniques and training on bigger datasets.

Key Achievements:

1. Successfully Integrated Machine Learning with Web Application Firewall (WAF):
Web application firewall (WAF) and machine learning were successfully integrated. Trained machine learning models were used to create an intelligent WAF that could recognize and react to unusual user activity.

2. Developed Dual Web Applications (Vulnerable & Secure):
Created two web applications, one secure and one vulnerable:
established two separate environments: one for actively stopping and rerouting questionable activity, and another for tracking user behavior.

3.Real-Time Behavior Monitoring & Logging: A system was put in place to monitor user entry and exit times as well as behavioral trends. All data was effectively stored in MongoDB for analysis.

4.Automatic Threat Detection & Response: Based on behavioral analysis and machine learning predictions, this feature allowed the secure online application to restrict or reroute users executing possible assaults.

5.Interactive Dashboard & Reporting: Interactive Dashboard & Reporting: A visual dashboard with options to export logs and reports was created that shows real-time user activity and threat warnings based on machine learning.

6. FUTURE ENHANCEMENTS

1. Integration of Deep Learning Models:
By using deep learning methods like RNNs or LSTM for sequential behavior analysis, threat detection may become more accurate and flexible.

2. Real-Time Alerting System:
When a significant anomaly or attack is discovered, administrators may be promptly notified by email, SMS, or push notifications.

3. Multi-Platform Support:
Provide cross-platform security by expanding the firewall's compatibility with mobile apps and APIs.

4.User Behavior Profiling:
Create thorough user profiles over time to identify minute behavioral changes that can point to compromised accounts or insider threats.

5. Self-Learning System:
To keep ahead of new attack patterns, let the system learn from fresh data continually and update the model automatically without the need for human retraining.

REFERENCES

[1] OWASP Top 10 Web Application Security Risks. Retrieved from https://owasp.org/www-project-top-ten/
[2] Scikit-learn Documentation. Retrieved from https://scikit-learn.org
[3] MongoDB Documentation. Retrieved from https://www.mongodb.com/docs/
[4] Python Flask Documentation. Retrieved from https://flask.palletsprojects.com
[5] TensorFlow Documentation. Retrieved from https://www.tensorflow.org
[6] PyMongo Documentation. Retrieved from https://pymongo.readthedocs.io
[7] Introduction to Machine Learning with Python by Andreas C. Müller & Sarah Guido
[8] Behavior-based Anomaly Detection Research Paper. Retrieved from https://ieeexplore.ieee.org/document/

[9] Real-Time Anomaly Detection System Using ML Techniques. Retrieved from https://www.sciencedirect.com

[10] A Study on Web Application Firewall with ML Support – ResearchGate. Retrieved from https://www.researchgate.net

[11] Web Application Firewall Based on Anomaly Detection using Deep Learning Authors: Sezer Toprak, Ali Gökhan Yavuz Published: 2022, Acta Infologica

[12] Leveraging Deep Neural Networks for Anomaly-Based Web Application Firewall Authors: Ali Moradi Vartouni, Mohammad Teshnehlab, Saeed Sedighian Kashi Published: 2019, IET Information Security

[13] ModSecurity Handbook – Ivan Ristić https://www.feistyduck.com/books/modsecurity-handbook/

[14] Machine Learning for Anomaly Detection – Chandola et al. https://www.cs.umn.edu/sites/cs.umn.edu/files/2021-10/AnomalyDetection.pdf

[15] **A** Survey on Web Application Firewall – IJARCCE https://www.ijarcce.com/upload/2018/february-18/IJARCCE%2016.pdf

[16] Detecting Malicious Web Requests Using Machine Learning – IEEE https://ieeexplore.ieee.org/document/9057766

[17] ML-Based WAF with LSTM – MDPI Sensors Journal https://www.mdpi.com/1424-8220/21/3/789

[18] Anomaly Detection in HTTP Traffic Using ML – Springer https://link.springer.com/article/10.1007/s10207-019-00472-8

[19] Web Application Firewall Evaluation Criteria – OWASP https://owasp.org/www-project-web-application-firewall-evaluation-criteria/

[20] Using ML to Detect Web Application Attacks – arXiv https://arxiv.org/abs/2007.10389

[21] Survey: ML Approaches for Cybersecurity – ACM https://dl.acm.org/doi/10.1145/3359789

[22] Anomaly-Based Intrusion Detection Using ML Techniques – Elsevier https://www.sciencedirect.com/science/article/pii/S1877050919311412

[23] Scikit-learn: ML in Python https://scikit-learn.org/stable/

[24] TensorFlow Security Use Cases https://www.tensorflow.org/security

[25] ML for Security: A Survey – arXiv https://arxiv.org/abs/1906.05799

[26] Using ML for HTTP Request Anomaly Detection –ResearchGate https://www.researchgate.net/publication/333497735

[27] Detecting Attacks with Autoencoders – IEEE https://ieeexplore.ieee.org/document/8620399

[28] OWASP ModSecurity Core Rule Set (CRS) https://coreruleset.org/

[29] Cloudflare WAF Architecture https://developers.cloudflare.com/waf/

[30] AWS WAF with ML https://docs.aws.amazon.com/waf/latest/developerguide/ml.html

[31] Azure Web Application Firewall https://learn.microsoft.com/en-us/azure/web-application-firewall/