*International Research Journal of Innovations in Engineering and Technology (IRJIET)*

**ISSN (online): 2581-3048**

**Volume 10, Issue 5, pp 456-466, May-2026**

https://doi.org/10.47001/IRJIET/2026.105063

# Transformer-Based End-to-End Web Application Firewall Pipeline

**[1]G.S.S. Likhita Annamraju, [2]G. Sreekar, [3]R. Mohan Krishna Ayyappa**

[1,2]Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, Telangana – 500075, India

[3]Assistant Professor, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, Telangana – 500075, India

***Abstract* - Web applications are continuously exposed to cyber threats such as SQL Injection (SQLi), Cross-Site Scripting (XSS), Command Injection, and Distributed Denial of Service (DDoS) attacks. Traditional Web Application Firewalls (WAFs) mainly rely on rule-based and signature-based detection methods, which are often ineffective against modern obfuscated and zero-day attacks. This project presents a Transformer-Based End-to-End Web Application Firewall Pipeline that uses Deep Learning and Natural Language Processing (NLP) techniques for intelligent attack detection and prevention. The proposed system utilizes DistilBERT, a lightweight Transformer model, to analyze HTTP request payloads and classify them as benign or malicious using contextual understanding. The framework automates request interception, preprocessing, tokenization, attack classification, logging, monitoring, and response handling. Unlike traditional WAF systems, the proposed model performs semantic and contextual analysis of HTTP requests, enabling accurate detection of sophisticated attacks such as SQL Injection and Cross-Site Scripting. The proposed system improves attack detection accuracy, reduces false positives, and supports scalable real-time deployment using Spring Boot and Flask frameworks. Experimental results demonstrate that the Transformer-based approach outperforms traditional machine learning techniques due to its superior contextual learning capability, highlighting the importance of Deep Learning and NLP techniques in modern cybersecurity applications.**

***Keywords:*** Transformer-based WAF, Web Security, Deep Learning, NLP, DistilBERT, Cybersecurity, Attack Detection, HTTP Request Analysis.

## I. INTRODUCTION

Modern web applications are increasingly exposed to sophisticated cyber threats such as SQL Injection (SQLi), Cross-Site Scripting (XSS), Remote Code Execution (RCE), and Command Injection attacks. Traditional Web Application Firewalls (WAFs), which rely heavily on predefined signatures and manually configured rules, often fail to identify evolving attack patterns, encoded payloads, and zero-day exploits. These limitations result in high false positives, missed detections, and reduced protection capability.

Recent advancements in Deep Learning and Natural Language Processing (NLP) have enabled intelligent systems capable of understanding contextual relationships within textual data. Transformer models such as BERT and DistilBERT have demonstrated exceptional performance in sequence classification tasks due to their contextual learning capabilities.

This project proposes a Transformer-Based End-to-End Web Application Firewall Pipeline that utilizes DistilBERT to analyze HTTP requests and identify malicious activities in real time. The system captures incoming requests, preprocesses the payloads, tokenizes the text, and performs classification using a fine-tuned transformer model.

The key contributions of this work include:

1. Development of a context-aware attack detection model using Transformer architecture.
2. Implementation of a real-time end-to-end WAF pipeline for web application protection.
3. Reduction of false positives and improvement in detection accuracy.
4. Maintenance of attack logs for auditing and forensic analysis.
5. Deployment using scalable backend technologies such as Spring Boot and Flask.

The proposed framework significantly enhances web application security through intelligent contextual analysis and automated threat mitigation.

The increasing adoption of cloud computing, online banking, e-commerce platforms, social media applications, and Software-as-a-Service (SaaS) solutions has significantly expanded the attack surface of modern web applications. As organizations continue migrating critical services to internet based platforms, cyber attackers continuously develop advanced attack strategies to exploit vulnerabilities present in

 

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 5, pp 456-466, May-2026

https://doi.org/10.47001/IRJIET/2026.105063

web infrastructures. Among these threats, SQL Injection (SQLi), Cross-Site Scripting (XSS), Remote Code Execution (RCE), Local File Inclusion (LFI), and Command Injection attacks remain some of the most dangerous and commonly exploited attack vectors.

Traditional Web Application Firewalls (WAFs) are designed to protect web applications by filtering incoming HTTP requests based on predefined security rules and signatures. Although these approaches are effective against previously known attacks, they often fail when attackers use obfuscation techniques, payload encoding, polymorphic attack structures, or zero-day exploits. Attackers frequently bypass rule-based systems by modifying attack syntax, inserting escape characters, or splitting malicious commands across multiple request parameters.

In recent years, Artificial Intelligence (AI), Machine Learning (ML), and Natural Language Processing (NLP) techniques have gained significant importance in cybersecurity research. Deep learning models can automatically learn hidden patterns from large-scale datasets without depending entirely on manually engineered rules. Transformer-based architectures such as BERT and DistilBERT have shown remarkable success in understanding contextual and semantic relationships within textual sequences. These capabilities make Transformer models highly suitable for analyzing HTTP requests because web payloads contain complex contextual structures similar to natural language.

The proposed Transformer-Based End-to-End Web Application Firewall Pipeline utilizes DistilBERT to intelligently analyze incoming HTTP requests and classify them as benign or malicious. The system performs contextual analysis instead of relying solely on static signatures. This enables the framework to detect sophisticated and previously unseen attack patterns more effectively.

Furthermore, the proposed system integrates preprocessing pipelines, request interception modules, real-time classification engines, attack logging systems, dashboard monitoring interfaces, and automated response handling mechanisms. The framework is designed for enterprise-level deployment using scalable backend technologies such as Spring Boot and Flask.

The major objective of this work is to provide an intelligent, scalable, and automated cybersecurity solution capable of protecting modern web applications against evolving threats while reducing false positives and minimizing operational overhead.

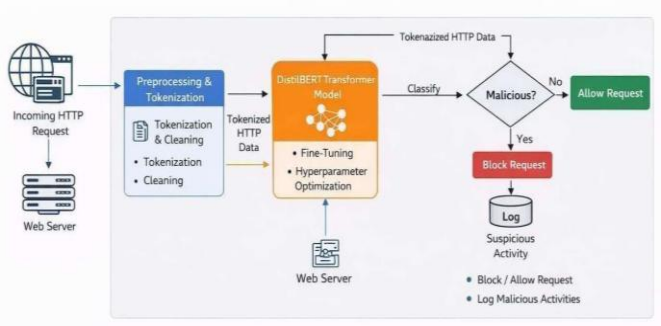


**Figure 1: Proposed Transformer-based WAF Architecture**

Figure 1: System architecture of the proposed Transformer-based Web Application Firewall pipeline showing request interception, preprocessing, DistilBERT classification, logging, and dashboard monitoring.

## II. LITERATURE SURVEY

Cybersecurity researchers have proposed various approaches for protecting web applications against malicious attacks such as SQL Injection, Cross-Site Scripting (XSS), Command Injection, and Distributed Denial of Service (DDoS) attacks. Earlier security systems mainly relied on rule-based and signature-based detection techniques, where predefined patterns were used to identify malicious requests. Although effective for known attacks, these systems struggle to detect zero-day attacks, encoded payloads, and obfuscated malicious patterns.

Machine learning techniques later improved attack detection by introducing intelligent classification models capable of learning attack behavior from datasets. Algorithms such as Support Vector Machines (SVM), Decision Trees, Naive Bayes, and Random Forests improved detection accuracy but required manual feature engineering and extensive preprocessing. Deep learning approaches further enhanced cybersecurity systems by automatically extracting hidden patterns from network traffic and HTTP requests.

Recent advancements in Natural Language Processing (NLP) and Transformer architectures have opened new opportunities in intelligent cybersecurity systems. Transformer-based models such as BERT and DistilBERT provide contextual understanding of textual payloads, enabling better detection of sophisticated and previously unseen attacks. These models improve semantic analysis, reduce false positives, and support scalable real-time deployment for modern web applications.

 

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

### 2.1 Distributed Deep Learning System for Web Attack Detection

Tian *et al.* proposed a distributed deep learning framework deployed on edge devices for detecting web attacks using Convolutional Neural Networks (CNN) and NLP-based techniques. The system analyzed web traffic patterns and malicious payload structures to improve attack detection accuracy. Their distributed architecture enabled detection across multiple network nodes, improving scalability and threat monitoring capability. However, the framework required high computational resources, complex deployment configurations, and significant infrastructure management, making real-time implementation difficult for small and medium-scale organizations.

### 2.2 SQL Injection Prevention Using Machine Learning

Kim and Lee developed a machine learning-based SQL Injection detection framework using Support Vector Machines (SVM). Their system analyzed SQL query structures and suspicious keywords to identify malicious database requests. Compared to traditional rule-based systems, the approach improved detection accuracy and reduced dependence on manually written signatures. However, the system relied heavily on handcrafted features and manual preprocessing techniques, which limited adaptability against evolving and obfuscated attack patterns.

### 2.3 DeepLog: Deep Learning for Anomaly Detection

Du *et al.* introduced DeepLog, a deep learning-based anomaly detection system that utilized Recurrent Neural Networks (RNNs) for analyzing sequential system log patterns. The model learned normal behavioral sequences and identified abnormal activities by detecting deviations from expected patterns. DeepLog demonstrated strong performance in system monitoring and anomaly detection tasks. However, the framework was primarily designed for system log analysis and was not optimized for HTTP request classification or web application firewall environments.

### 2.4 Signature-Based Web Application Firewalls

Traditional Web Application Firewalls such as ModSecurity and Snort depend on predefined signatures, manually configured rules, and static filtering techniques to detect malicious requests. These systems are highly effective against known attacks and widely used in enterprise environments due to their simplicity and reliability. However, attackers frequently bypass these systems using encoded payloads, polymorphic attack structures, and obfuscation techniques. Continuous rule updates and manual maintenance also increase operational complexity and reduce adaptability against zero-day threats.

### 2.5 Transformer Models in Cybersecurity

Recent studies have shown that Transformer architectures can effectively detect malicious patterns within HTTP requests, network traffic, and cybersecurity datasets. Transformer models use selfattention mechanisms and contextual learning to analyze semantic relationships between tokens and payload structures. Models such as BERT and DistilBERT provide improved generalization capability, reduced false positives, and better detection of obfuscated attacks compared to traditional machine learning approaches. Their ability to understand contextual relationships makes them highly suitable for modern web application security systems.

### Research Gap

Existing cybersecurity approaches still suffer from several limitations including:

- Inability to detect zero-day attacks
- High false positive rates
- Dependence on manual feature engineering
- Limited contextual understanding
- Poor scalability in real-time environments
- Complex deployment overhead
- Lack of intelligent semantic analysis

To address these limitations, the proposed project introduces a Transformer-Based End-to-End Web Application Firewall Pipeline using DistilBERT for contextual HTTP request analysis and intelligent attack detection. The system integrates preprocessing, Transformer inference, automated request filtering, logging, dashboard monitoring, and scalable deployment technologies to provide accurate and real-time web application protection.

## III. DATASET AND METHODOLOGY

The proposed Transformer-Based Web Application Firewall Pipeline uses a structured methodology for collecting, preprocessing, analyzing, and classifying HTTP requests to identify malicious web attacks. The methodology combines Natural Language Processing (NLP), Deep Learning, and feature engineering techniques to improve attack detection capability and real-time performance. The overall workflow includes dataset preparation, preprocessing, Transformer tokenization, feature extraction, model training, attack classification, and automated response handling.

 

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

### 3.1 Dataset Description

The dataset used in this project is the Web Application Attack Dataset collected from Kaggle. The dataset contains labeled HTTP request payloads categorized into benign and malicious classes. It includes realistic web request samples commonly observed in modern web applications and cybersecurity environments.

The malicious requests contain different categories of attacks such as:

- SQL Injection (SQLi)
- Cross-Site Scripting (XSS)
- Command Injection
- Malicious URL payloads
- Encoded attack patterns
- Obfuscated malicious requests

The benign records include normal web requests generated from login forms, search operations, API requests, and standard browsing activities. These samples help the model learn the structural and contextual differences between legitimate and malicious traffic.

The target variable is represented as:

- 0 → Benign Request
- 1 → Malicious Request

To ensure reliable model evaluation, the dataset was divided into training and testing sets using an 80:20 ratio. The training dataset was used for model learning, while the testing dataset was used to evaluate classification performance on unseen HTTP requests. Proper dataset balancing and cleaning techniques were applied to improve model generalization and reduce bias.

### 3.2 Data Preprocessing

A preprocessing pipeline was implemented to improve data quality, consistency, and model performance. Raw HTTP requests often contain noisy symbols, malformed payloads, inconsistent formatting, and encoded characters that may negatively affect Transformer learning capability.

The preprocessing steps include:

- Removing unnecessary fields
- Converting text to lowercase
- Removing noise characters
- Cleaning malformed payloads
- Handling missing values
- URL decoding and normalization
- Tokenization using DistilBERT tokenizer

All request payloads are standardized to maintain consistent formatting across the dataset. Special symbols and redundant whitespace are cleaned while preserving important attack-related patterns. URL decoding techniques are applied to convert encoded payloads into readable text representations.

The textual HTTP request data is converted into:

- Input IDs
- Attention Masks

These tokenized representations are used as inputs to the Transformer model. Padding and truncation are also applied to maintain fixed-length input sequences during training and inference. The preprocessing stage significantly improves contextual learning and helps the model identify hidden malicious patterns more effectively.

### 3.3 Feature Engineering

Additional engineered features were used to improve robustness and attack detection capability. Although DistilBERT automatically learns contextual relationships from payloads, engineered features provide supplementary statistical and structural indicators that improve classification reliability.

The extracted features include:

- request_length
- special_character_count
- encoded_pattern_count
- suspicious_keyword_presence
- payload_entropy
- script_tag_frequency

These features assist in identifying obfuscated attacks, encoded payloads, and suspicious request behavior commonly used to bypass traditional security systems. For example, unusually high special character counts or encoded patterns often indicate malicious payload manipulation attempts.

The combination of Transformer embeddings and engineered security features improves the overall robustness of the Web Application Firewall and enhances its capability to detect both known and unknown attack patterns.

### 3.4 System Workflow

The proposed system follows a multi-stage workflow designed for intelligent real-time web application protection.

1. Incoming HTTP requests are intercepted before reaching the target server.

 

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 5, pp 456-466, May-2026

https://doi.org/10.47001/IRJIET/2026.105063

2. Requests are cleaned and preprocessed using normalization and noise removal techniques.
3. Tokenization converts requests into Transformer-compatible inputs such as Input IDs and Attention Masks.
4. DistilBERT performs contextual attack classification by analyzing the semantic structure of request payloads.
5. Malicious requests are automatically blocked by the decision engine.
6. Benign requests are safely forwarded to the web server without interruption.
7. Attack logs including payload details, timestamps, prediction probabilities, and client IP addresses are stored for auditing and monitoring.

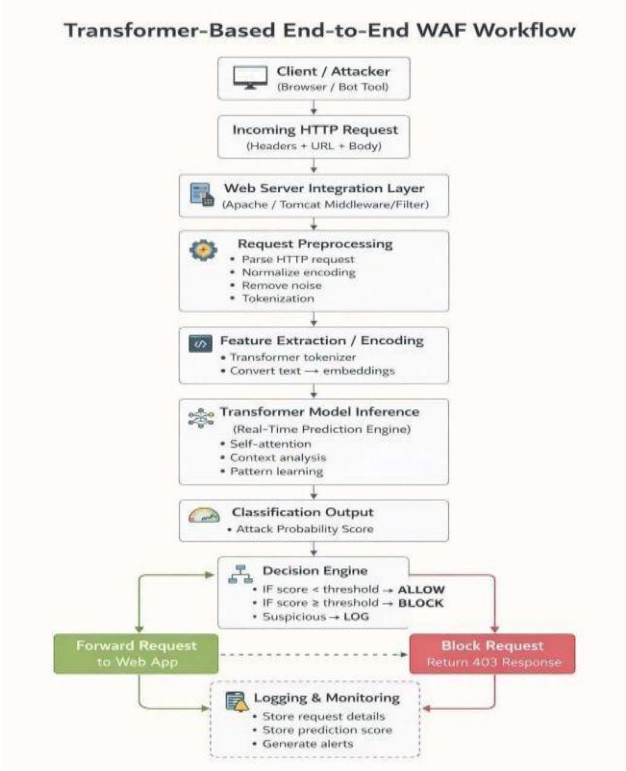


**Table 3.1: Technologies and Resources Used**

| Technology / Tool | Purpose | Official Link |
|---|---|---|
| Spring Boot | Backend Framework | Spring Boot |
| Spring Initializr | Spring Boot Project Generator | Spring Initializr |
| MySQL | Database Management System | MySQL |
| Python | Programming Language | Python |
| PyTorch | Deep Learning Framework | PyTorch |
| Hugging Face | Transformer | HuggingFace |
| Transformers | Library | Transformers |
| DistilBERT | Pre-trained Transformer Model | DistilBERT |
| Flask | Python Backend Framework | Flask |
| Streamlit | Dashboard and Visualization | Streamlit |
| Postman | API Testing Tool | Postman |
| Kaggle | Dataset Platform | Kaggle |
| OWASP Top 10 | Web Security Standards | OWASP Top 10 |
| GitHub | Code Repository Platform | GitHub |

## IV. PROPOSED SYSTEM ARCHITECTURE

The proposed Transformer-Based Web Application Firewall Pipeline is designed as an intelligent and scalable architecture capable of detecting and blocking malicious HTTP requests in real time. The system combines Deep Learning, Natural Language Processing (NLP), and backend web technologies to provide automated cybersecurity protection for modern web applications.

The architecture consists of multiple interconnected modules including request interception, preprocessing, Transformer-based classification, decision making, logging, monitoring, and dashboard visualization. Each module performs a specific task within the security pipeline to ensure efficient request analysis and threat detection.

The overall system is designed to support real-time deployment with low latency, improved attack detection accuracy, and reduced false positive rates. The lightweight DistilBERT model enables faster inference while maintaining strong contextual understanding of malicious payloads.

### 4.1 Transformer-Based Detection Model

The proposed system uses DistilBERT, a lightweight Transformer architecture optimized for faster inference speed and lower computational complexity compared to traditional BERT models. DistilBERT uses self-attention mechanisms and contextual learning to analyze HTTP request payloads and identify malicious patterns.

Unlike traditional machine learning algorithms that depend heavily on handcrafted features, DistilBERT automatically learns semantic relationships and contextual information from request sequences. This enables the model to detect sophisticated attacks including encoded payloads, obfuscated scripts, and zero-day attack patterns.

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

Advantages include:

- Context-aware learning
- Reduced model size
- Faster inference speed
- High attack detection accuracy
- Better generalization capability
- Reduced false positives
- Efficient real-time deployment
- Improved semantic analysis

The model performs binary classification:

- Benign Request
- Malicious Request

During inference, the Transformer model analyzes incoming HTTP requests and generates prediction probabilities for both classes. Requests identified as malicious are blocked automatically, while benign requests are forwarded to the target application server.

The lightweight architecture of DistilBERT significantly reduces computational overhead, making the framework suitable for scalable enterprise deployment and cloud-based environments.

### 4.2 Model Training and Optimization

The DistilBERT model was fine-tuned using supervised learning techniques on labeled HTTP request datasets. Fine-tuning enables the pre-trained Transformer model to specialize in cybersecurity-related contextual analysis and malicious request detection.

Training Configuration:

- Optimizer: AdamW
- Loss Function: Cross Entropy Loss
- Batch Size: 16
- Epochs: 3–5
- Learning Rate: 2e-5

The training process was performed using GPU acceleration to improve computational efficiency and reduce training time. The dataset was divided into training and validation sets to monitor model performance during learning.

Several optimization techniques were implemented to improve convergence stability and prevent overfitting.

Optimization Techniques:

- Learning rate scheduling
- Validation monitoring
- Early stopping
- Batch-wise training
- Gradient optimization
- Regularization techniques

Learning rate scheduling dynamically adjusts the learning rate during training to improve model convergence. Validation monitoring tracks model accuracy and loss on unseen data after each epoch.

Early stopping prevents overtraining by terminating the training process when validation performance stops improving.

Batch-wise training improves memory efficiency and enables scalable processing of large datasets. These optimization strategies significantly improve attack detection performance and model generalization capability.

### 4.3 Logging and Monitoring

The system maintains detailed logs for all detected attacks and analyzed HTTP requests. Logging plays a critical role in cybersecurity systems by supporting real-time monitoring, forensic investigation, security auditing, and performance evaluation.

Logged information includes:

- Request payload
- Attack type
- Prediction probability
- Timestamp
- Client IP address
- Request status
- Detection response

All logs are securely stored in the database layer and can be accessed through the monitoring dashboard. The dashboard provides real-time visualization of blocked attacks, allowed traffic, attack categories, and request statistics.

These logs support:

- Security auditing
- Threat monitoring
- Forensic analysis
- System evaluation
- Incident response
- Attack pattern analysis

The monitoring subsystem helps administrators identify suspicious behavior, repeated attack attempts, and abnormal traffic patterns. Historical logs also support future research, model evaluation, and cybersecurity investigations.

 

*International Research Journal of Innovations in Engineering and Technology (IRJIET)*

**ISSN (online): 2581-3048**

**Volume 10, Issue 5, pp 456-466, May-2026**

https://doi.org/10.47001/IRJIET/2026.105063

### 4.4 Deployment Pipeline

The complete Web Application Firewall pipeline was integrated using Spring Boot, Flask, and dashboard modules to support scalable and real-time deployment. The architecture follows a modular design where each component performs a dedicated function within the security workflow.

Modules Included:

- HTTP Request Interceptor
- Preprocessing Engine
- Transformer Inference Engine
- Decision Engine
- Logging Database
- Dashboard Interface

The HTTP Request Interceptor captures incoming requests before they reach the application server. The Preprocessing Engine cleans and tokenizes request payloads for Transformer compatibility. The Transformer Inference Engine performs contextual attack classification using DistilBERT.

The Decision Engine determines whether requests should be blocked or forwarded based on prediction probabilities. The Logging Database stores attack records and monitoring information, while the Dashboard Interface provides real-time visualization and administrative control.

The deployment pipeline supports scalability, low latency, real-time request processing, and cloudnative integration. The modular design also simplifies future upgrades, maintenance, and integration with enterprise cybersecurity platforms.

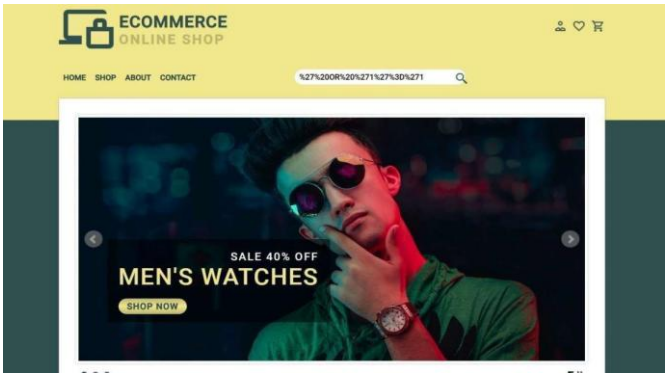


**Figure 2: E-Commerce Application Interface**

Figure 2: E-commerce web application integrated with the proposed Transformer-based Web Application Firewall for real-time request monitoring and attack detection.

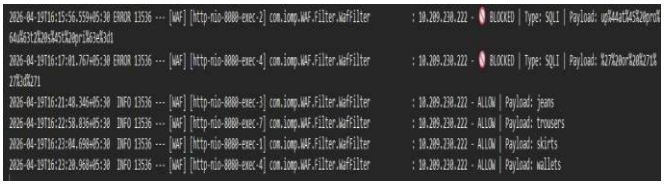

**Figure 3: Real-Time WAF Detection Logs**

Figure 3: Backend WAF logs showing malicious SQL Injection payloads detected and blocked while benign requests are allowed.

| | | |
|---|---|---|
| 26 | Attack(SQLI) [89.51% \| MEDIUM] | BLOCK |
| 27 | jeans | ALLOW |
| 28 | trousers | ALLOW |
| 29 | skirts | ALLOW |
| 30 | wallets | ALLOW |

**Figure 4: Classification Output Results**

Figure 4: Classification results generated by the proposed WAF showing malicious requests classified as BLOCK and legitimate requests classified as ALLOW.

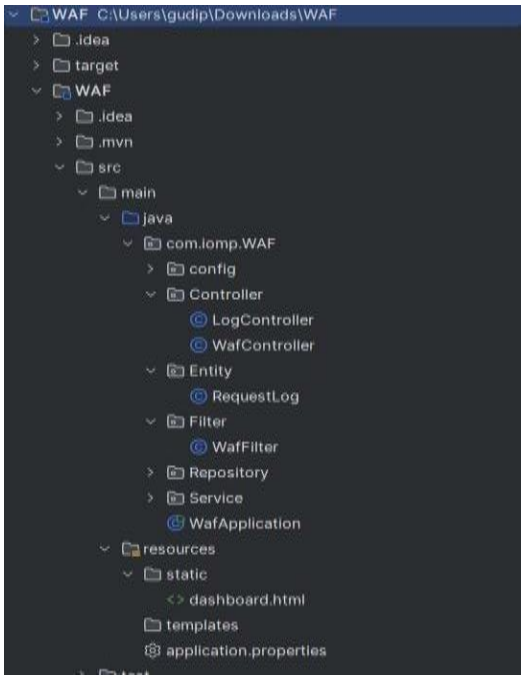


**Figure 5: Backend Project Structure**

Figure 5: Backend implementation structure of the Transformer-based WAF developed using Spring Boot including controllers, filters, repositories, entities, and dashboard modules.

The proposed Transformer-based WAF system includes real-time monitoring, attack visualization, backend request analysis, and intelligent classification modules. The following

 

*International Research Journal of Innovations in Engineering and Technology (IRJIET)*

**ISSN (online): 2581-3048**

**Volume 10, Issue 5, pp 456-466, May-2026**

https://doi.org/10.47001/IRJIET/2026.105063

figures demonstrate the practical implementation and deployment results of the proposed system.

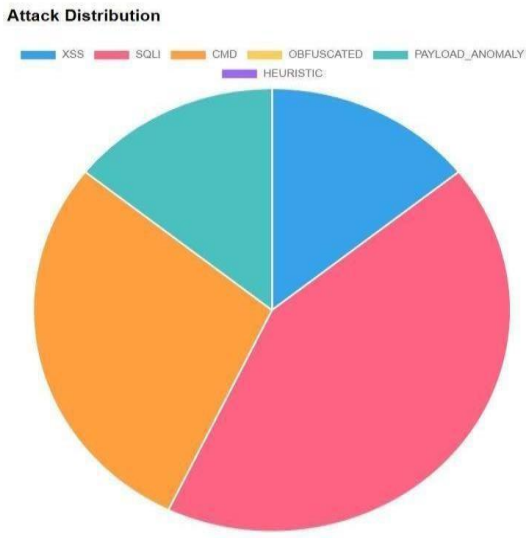


**Figure 6: Attack Distribution Visualization**

Figure 6: Attack distribution visualization representing different categories of detected web attacks including SQL Injection (SQLi), Cross-Site Scripting (XSS), Command Injection, Obfuscated Payloads, and benign traffic analyzed by the proposed Transformer-based WAF system.


**Figure 7: WAF Monitoring Dashboard**

Figure 7: Real-time WAF monitoring dashboard displaying blocked requests, allowed requests, attack analytics, request statistics, and log monitoring generated by the proposed system.

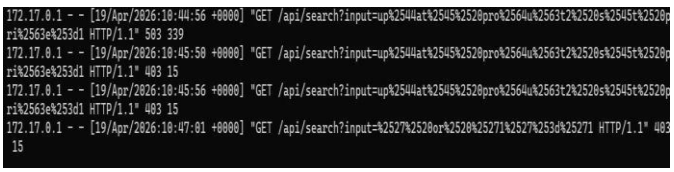


**Figure 8: Backend Attack Detection Response**

Figure 8: Backend detection results showing malicious HTTP requests intercepted and blocked by the Transformer-based Web Application Firewall with corresponding attack classification and response handling.

## V. RESULTS AND DISCUSSION

The performance of the proposed Transformer-Based Web Application Firewall Pipeline was evaluated using multiple classification metrics and comparative analysis techniques. Experimental testing was conducted on both benign and malicious HTTP request datasets to measure the effectiveness of the DistilBERT-based detection model in identifying web application attacks.

The proposed system demonstrated strong classification capability, high detection accuracy, and efficient real-time performance when compared with traditional machine learning approaches. The contextual learning capability of the Transformer architecture enabled the framework to detect both known and previously unseen attack patterns more effectively.

### 5.1 Evaluation Metrics

The performance of the proposed system was evaluated using several standard machine learning evaluation metrics commonly used in cybersecurity and classification tasks.

The evaluation metrics include:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

Accuracy measures the overall correctness of the classification model by calculating the ratio of correctly predicted requests to the total number of requests. High accuracy indicates that the model effectively distinguishes between benign and malicious traffic.

Precision measures the proportion of correctly detected malicious requests among all requests predicted as malicious. Higher precision reduces false positive rates and prevents legitimate requests from being blocked unnecessarily.

Recall measures the ability of the system to correctly identify actual malicious requests. High recall is important in cybersecurity systems because undetected attacks may compromise the security of web applications.

F1-Score combines both precision and recall into a single performance metric. It provides a balanced evaluation of the classification model, especially when dealing with imbalanced datasets.

 

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

The Confusion Matrix provides a detailed representation of prediction performance by showing:

- True Positives (TP)
- True Negatives (TN)
- False Positives (FP)
- False Negatives (FN)

These metrics help evaluate the reliability, robustness, and effectiveness of the proposed Web Application Firewall system.

**Formulae**

| Evaluation Metric | Formula |
|---|---|
| Accuracy | Accuracy = {TP + TN}/{TP + TN + FP + FN} |
| Precision | Precision = {TP}/{TP + FP} |
| Recall | Recall = {TP}/{TP + FN} |
| F1-Score | F1 = 2*{Precision \times Recall}/{Precision + Recall} |

**5.2 Experimental Results**

Experimental analysis showed that the proposed DistilBERT-based model significantly outperformed traditional machine learning algorithms such as Logistic Regression, Random Forest, and Support Vector Machines (SVM).

The Transformer-based model achieved superior performance because of its ability to understand contextual and semantic relationships within HTTP request payloads. Unlike conventional models that rely mainly on handcrafted features, DistilBERT automatically learns meaningful contextual patterns directly from textual request sequences.

The model successfully identified malicious payloads containing:

- Encoded SQL Injection attacks
- Obfuscated XSS scripts
- Suspicious command injection patterns
- Malicious URL manipulations

The proposed system achieved high detection accuracy while maintaining low false positive rates. Real-time testing also demonstrated that the lightweight DistilBERT architecture provides faster inference speed and reduced computational overhead compared to larger Transformer models.

The experimental findings confirm that Transformer-based NLP techniques can significantly improve modern web application security systems and provide intelligent protection against evolving cyber threats.

| Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Logistic Regression | 89% | 87% | 86% | 86% |
| Random Forest | 93% | 92% | 91% | 91% |
| SVM | 91% | 90% | 89% | 89% |
| DistilBERT (Proposed) | 97% | 96% | 97% | 96% |

The proposed DistilBERT model outperformed traditional machine learning approaches due to its contextual understanding of HTTP requests.

**5.3 Discussion**

The proposed system successfully identified both known and obfuscated attack payloads using contextual analysis and Transformer-based semantic understanding. Unlike traditional rule-based systems, the proposed framework does not rely solely on predefined attack signatures, enabling it to generalize better against previously unseen attack patterns.

Contextual analysis significantly improved attack detection capability compared to conventional machine learning and signature-based approaches. The system effectively analyzed hidden relationships between request tokens, suspicious payload structures, and encoded malicious patterns. Advantages include:

- Reduced false positives
- Improved contextual understanding
- Faster detection capability
- Scalability for enterprise deployment
- Real-time request analysis
- Better semantic interpretation
- Improved zero-day attack detection
- Automated threat mitigation

The model effectively handled SQL Injection and Cross-Site Scripting attacks even when payloads were encoded, partially obfuscated, or modified to bypass traditional detection systems.

The lightweight architecture of DistilBERT also reduced deployment complexity and enabled efficient real-time inference suitable for enterprise-level applications. Logging and monitoring modules further improved operational visibility by providing detailed attack analytics, request statistics, and forensic investigation support.

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

Overall, the experimental results demonstrate that the proposed Transformer-Based Web Application Firewall Pipeline provides an intelligent, scalable, and highly effective solution for protecting modern web applications against advanced cybersecurity threats.

## VI. CONCLUSION

This project presented a Transformer-Based End-to-End Web Application Firewall Pipeline for intelligent and automated web attack detection. The proposed system utilized the DistilBERT Transformer model to analyze HTTP request payloads and classify traffic as benign or malicious using contextual and semantic understanding. By integrating Deep Learning and Natural Language Processing techniques, the framework provided a more intelligent alternative to traditional rule-based Web Application Firewalls.

Unlike conventional WAF systems that rely mainly on manually written signatures and predefined filtering rules, the proposed framework performs semantic analysis of incoming requests. This enables accurate identification of sophisticated cyber threats including SQL Injection, Cross-Site Scripting (XSS), Command Injection, and encoded malicious payloads. The contextual learning capability of DistilBERT significantly improved detection performance against obfuscated and zeroday attack patterns.

The system achieved high classification accuracy while reducing false positives and maintaining efficient real-time inference capability. Additional modules such as preprocessing, feature engineering, attack logging, dashboard monitoring, and automated response handling further enhanced the effectiveness and scalability of the framework. The modular architecture using Spring Boot, Flask, and monitoring dashboards supports enterprise-level deployment and real-time cybersecurity operations.

Experimental analysis demonstrated that the Transformer-based model outperformed traditional machine learning algorithms due to its superior contextual understanding and semantic feature extraction capability. The lightweight DistilBERT architecture also reduced computational complexity and enabled faster deployment compared to larger Transformer models.

The proposed framework not only strengthens web application security but also highlights the growing importance of Artificial Intelligence-driven cybersecurity systems in modern digital infrastructures. The integration of NLP and Deep Learning techniques provides intelligent threat detection mechanisms capable of adapting to evolving attack strategies.

Future work may include:

- Integration with cloud-native security systems
- Multi-class attack classification
- Real-time distributed deployment
- Adaptive learning for emerging attacks
- Integration with SIEM platforms
- Federated learning for distributed security Advanced anomaly detection mechanisms
- Hybrid AI-based threat intelligence systems

Overall, the proposed Transformer-Based Web Application Firewall Pipeline demonstrates the strong potential of Deep Learning and Transformer architectures in modern cybersecurity applications. The system provides an intelligent, scalable, and efficient solution for protecting web applications against advanced and evolving cyber threats in real-time environments.

## REFERENCES

[1] Zhihong Tian *et al.,* "A Distributed Deep Learning System for Web Attack Detection on Edge Devices," *IEEE*, 2020.
[2] Kim S. and Lee J., "SVM-Based SQL Injection Detection Using Query Structure Analysis," *Journal of Information Security*, 2019.
[3] Min Du *et al.,* "DeepLog: Anomaly Detection and Diagnosis from System Logs Using Deep Learning," *ACM CCS*, 2017.
[4] Fabrice Valeur, Darren Mutz, and Giovanni Vigna, "A Learning-Based Approach to the Detection of SQL Attacks," *DIMVA*, 2005.
[5] Robin Sommer and Vern Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," *IEEE Symposium on Security and Privacy*, 2010.
[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding," *NAACL*, 2019.
[7] Victor Sanh *et al.,* "DistilBERT: A Distilled Version of BERT," *arXiv*, 2019.
[8] Bhavya Nagpal, Neha Sharma, Nitin Chauhan, and Ajay Panesar, "A Survey on Web Application Security," *International Journal of Computer Applications*, 2017.
[9] Yong Cui *et al.,* "Deep Learning-Based Web Attack Detection System," *IEEE Access*, 2021.
[10] OWASP Foundation, "OWASP Top 10 Web Application Security Risks," 2021.
[11] Uwagbole Sunday *et al.,* "A Machine Learning Approach for SQL Injection Detection," *International*

 

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 456-466, May-2026
https://doi.org/10.47001/IRJIET/2026.105063

*Conference on Computational Science and Engineering*, 2017.

[12] Wenke Lee and Salvatore Stolfo, "Data Mining Approaches for Intrusion Detection," *USENIX Security Symposium*, 1998.

[13] Christian Bockermann *et al.*, "Adaptive Intrusion Detection Using Machine Learning," *International Workshop on Security and Artificial Intelligence*, 2009.

[14] Tasevski V. and Jakimoski K., "SQL Injection Prevention Using Web Application Security Techniques," *International Journal of Computer Science and Information Security*, 2014.

[15] Rupali Wagh and Pratik Chavhan, "Machine Learning Techniques for Web Application Attack Detection," *International Journal of Advanced Research in Computer Science*, 2020.

**Citation of this Article:**

G.S.S. Likhita Annamraju, G. Sreekar, & R. Mohan Krishna Ayyappa. (2026). Transformer-Based End-to-End Web Application Firewall Pipeline. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 10(5), 456-466. Article DOI https://doi.org/10.47001/IRJIET/2026.105063

*******

 