

Extracting Attack Pattern From WAF Logs and CTIs Using Contrastive Semantic Learning

Sheng-Shan Chen¹, Graduate Student Member, IEEE, Ren-Hung Hwang², Senior Member, IEEE, Ying-Dar Lin³, Fellow, IEEE, Tun-Wen Pai⁴, Senior Member, IEEE, and Chin-Yu Sun⁵, Member, IEEE

Abstract—Web Application Firewalls (WAFs) are widely deployed to protect web services, but their rule-based design provides limited visibility into attacker intent. WAF logs consist primarily of low-level HTTP artifacts that lack the behavioral context required for effective threat analysis. To address this limitation, we propose the first automated framework that maps WAF logs to MITRE ATT&CK techniques through CTI-grounded semantic learning. The approach integrates structure-aware Monte Carlo Tree Search-based payload generation, CodeBERT-driven contrastive learning for attack classification, and cyber threat intelligence (CTI) alignment for TTP retrieval. The framework is evaluated on over 714,000 WAF logs derived from validated attack payloads across eight attack types, generated within a controlled environment using ModSecurity and OWASP Core Rule Set (CRS). Experimental results demonstrate 99.38% multi-class classification F1 score and identification of 206 unique ATT&CK techniques. Compared with a Rule-ID Heuristic baseline derived from OWASP CRS rule semantics, the proposed framework identifies $7.4\times$ more unique ATT&CK techniques and provides substantially broader TTP-level visibility. External validation on a real-world ModSecurity log dataset further demonstrates that the framework preserves reliable classification and retrieval performance beyond the controlled payload-generation setting.

Index Terms—Web application firewall (WAF), cyber threat intelligence (CTI), TTP identification, contrastive learning, Monte Carlo tree search (MCTS), semantic search.

I. INTRODUCTION

WEB applications have become critical infrastructure for modern digital services, making them prime targets for cyber attacks. Web Application Firewalls (WAFs) serve as the primary defense line to intercept malicious traffic before it impacts backend systems [1]. Among open-source solutions, ModSecurity combined with the OWASP Core Rule Set (CRS)

represents the industry standard, providing broad signature-based coverage for common vectors such as SQL injection (SQLi), Cross-Site Scripting (XSS), and file inclusion [2]. However, while WAFs excel at blocking malicious traffic, they face a fundamental limitation: a significant “semantic gap” exists between the low-level operational output of these systems and the high-level strategic intelligence required for effective threat analysis [3].

This semantic gap manifests in critical ways. WAF audit logs primarily capture syntactic features such as fragmented URIs, encoded payloads, and rule trigger metadata that reveal what was blocked but provide little insight into the adversary’s underlying intent or how the attack fits into broader adversarial campaigns. To derive meaningful threat context, analysts are forced to manually interpret fragmented HTTP requests, often obfuscated or mutated, a process that is labor-intensive, inconsistent, and increasingly unsustainable as attack surfaces expand [4]. Existing rule-based detection provides minimal taxonomical precision; a single payload often triggers generic alerts rather than specific attack classifications, leaving security teams without the behavioral context essential for strategic threat analysis.

Previous research has attempted to bridge this gap through anomaly detection, payload classification, and adaptive rule-learning techniques [5], [6], [7]. Although these methods have demonstrated promise in detecting known attack patterns, they share three critical limitations that prevent their effective deployment in operational environments.

First, most approaches treat payloads as unstructured text, failing to capture the deeper syntactic and semantic structures inherent in complex exploitation attempts. This oversight reduces resilience to novel or obfuscated attacks [8]. Second, existing datasets are frequently small, synthetic, or lack verified ground truth, limiting the ability of models to generalize to real-world adversarial behaviors [9], [10]. Third, and most critically, to the best of our knowledge, no prior work has established an automated mechanism that maps raw WAF telemetry to standardized MITRE ATT&CK techniques, a capability essential for aligning low-level WAF evidence with behavior-oriented threat intelligence [11].

Addressing these limitations requires a framework capable of: (1) overcoming the scarcity of high-quality, labeled attack data; (2) handling the long-tailed distribution of web attack behaviors; and (3) discovering latent semantic structures to identify unknown threats beyond fixed, supervised classes. Accomplishing these goals is non-trivial, as real-world WAF

Received 12 January 2026; revised 16 May 2026 and 13 September 2026; accepted 27 September 2026. Date of publication 30 September 2026; date of current version 9 October 2026. This work was supported in part by the National Institute of Cyber Security (NICS), Taiwan. The associate editor coordinating the review of this article and approving it for publication was A. Lahmadi. (Corresponding authors: Chin-Yu Sun; Ren-Hung Hwang.)

Sheng-Shan Chen is with the Department of Computer Science and Information Engineering, National Taipei University of Technology (NTUT), Taipei 106344, Taiwan, and also with CyCraft Technology Corporation, New Taipei City 220630, Taiwan.

Ren-Hung Hwang is with the Department of Computer Science and Information Engineering, Asia University, Taichung 41300, Taiwan, and also with the College of Artificial Intelligence, National Yang Ming Chiao Tung University, Tainan 300093, Taiwan (e-mail: rhhwang@asia.edu.tw).

Ying-Dar Lin is with the Department of Computer Science, National Yang Ming Chiao Tung University (NYCU), Hsinchu 300093, Taiwan.

Tun-Wen Pai and Chin-Yu Sun are with the Department of Computer Science and Information Engineering, National Taipei University of Technology (NTUT), Taipei 106344, Taiwan (e-mail: cysun@ntut.edu.tw).

Digital Object Identifier 10.1109/TNSM.2026.3738730

1932-4537 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on October 24, 2026 at 08:36:33 UTC from IEEE Xplore. Restrictions apply.

telemetry is noisy, highly redundant, and replete with malformed payloads that traditional supervised models struggle to utilize effectively.

To overcome these limitations, this study presents a novel attribution framework designed to transform raw WAF telemetry into high-fidelity threat intelligence. Our approach integrates three synergistic components: structure-aware payload generation with semantic verification, contrastive semantic learning for robust attack classification, and CTI-grounded retrieval for automated TTP mapping.

We first construct a controlled yet realistic evaluation environment using ModSecurity [12] (v3) and the OWASP CRS [13] (v3.3.5) deployed over the Damn Vulnerable Web Application (DVWA) [14]. This setup is not merely a testbed but a validation mechanism; by executing payloads against actual vulnerabilities, we ensure that our training data consists of ground-truth validated exploitations rather than theoretical heuristics.

Building upon this foundation, we introduce a structure-aware payload generation framework based on Monte Carlo Tree Search (MCTS). Extending the methodology originally proposed in AdvSQLi [15], which utilized MCTS specifically for SQLi, we adapt and scale this approach to encompass a broader spectrum of web threats. Unlike random fuzzing, this module synthesizes diverse, syntactically valid attack variants, effectively solving the data scarcity problem. The resulting data comprise 175,509 payload samples and semantic-verification pairs, together with 714,162 WAF logs spanning eight distinct traffic categories: SQLi, XSS, Command Injection (CI), Malicious File Upload (MFU), Cross-Site Request Forgery (CSRF), Local File Inclusion (LFI), Brute Force (BF), and Benign traffic.

The core of our proposed framework is a hybrid learning architecture that integrates Contrastive Learning based on CodeBERT (CodeBERT-CL) [16] with clustering of K-means [17]. While CodeBERT-CL learns robust and invariant representations of known attack types, the integration of K-means enables cluster-level assignment over the learned contrastive embedding space and helps assess whether unseen variants fall within the semantic regions of known attack families. This dual approach allows the model to classify known threats with high precision and to generalize to previously unseen variants and potential zero-day attacks that deviate from established clusters.

Finally, to bridge the semantic gap between WAF logs and threat intelligence, we introduce a cyber threat intelligence (CTI) alignment module that automatically converts WAF logs into TTP-style descriptions, enabling semantic retrieval across 18,088 CTI records. This module leverages large language models (LLMs) for knowledge expansion and embedding-based similarity search to identify the most relevant MITRE ATT&CK techniques.

Experimental results demonstrate that the proposed framework achieves a classification F1 score of 99.38% and identifies a total of 206 unique ATT&CK techniques. Crucially, even after applying a strict frequency threshold to isolate high-confidence behaviors, the framework retains 75 robust techniques, a $2.68\times$ improvement over the 28 techniques

inferred from OWASP CRS rule semantics, thus ensuring both broad visibility and operational reliability. To support this comparison, we introduce direct TTP attribution baselines, including an OWASP CRS Rule-ID Heuristic and a supervised SecureBERT2.0-based TTP classifier. The OWASP CRS Rule-ID Heuristic applies to only 13.59% of evaluated cases, while the SecureBERT2.0 classifier achieves only 0.1940 macro-F1 under direct single-label TTP classification [18]. In contrast, the proposed CTI-grounded retrieval module achieves a Top-3 accuracy of 86.35%. The CTI alignment module achieves an 87.10% top-5 retrieval recall, confirming its efficacy in automating threat hunting and incident response. Beyond the controlled DVWA-based evaluation, we also examine the trained framework on an independently collected thirty-day production ModSecurity dataset [19]. In this external setting, the CodeBERT-CL + K-means classifier achieves a 93.93% macro-F1 score for attack-type classification, while the CTI-grounded retrieval module reaches 82.00% Top-5 accuracy for TTP attribution.

The primary contributions of this study are summarized as follows:

- We propose the first framework for deriving standardized MITRE ATT&CK techniques from raw WAF logs through CTI-grounded semantic retrieval, bridging the gap between low-level telemetry and high-level threat intelligence.
- We introduce an MCTS-based generation method with semantic verification that synthesizes diverse valid malicious payloads, effectively addressing the challenge of data scarcity in training robust models.
- We develop a hybrid model combining CodeBERT-CL and K-means clustering. This architecture captures intrinsic attack semantics to robustly classify evasion variants while leveraging unsupervised learning to uncover potential unknown threats.
- We demonstrate an automated alignment module that enriches WAF logs with TTP descriptions, achieving large-scale semantic retrieval for security operations.
- We further validate the framework on an independently collected real-world ModSecurity dataset, demonstrating its effectiveness beyond the adversarial payload generation phase and under naturally occurring CRS rule-trigger distributions.

The remainder of this paper is structured as follows. Section II establishes the foundational background regarding WAF operations and the strategic role of CTI. Section III discusses related work in the fields of payload generation, attack classification, and TTP alignment. Section IV formalizes the research problem, specifically addressing the semantic gap between low-level WAF logs and actionable threat intelligence. Section V details the methodology of the proposed attribution framework. Section VI describes the system implementation, covering the ModSecurity and CRS environment, as well as the dataset construction process. Section VII presents the experimental setup and analyzes the empirical results. Section VIII discusses the study's implications, limitations, and

threats to validity. Finally, Section IX summarizes the study's contributions and outlines future research directions.

II. BACKGROUND

A. WAF

WAF constitutes the ubiquitous defense mechanism for modern web infrastructure, designed to inspect and filter Hypertext Transfer Protocol (HTTP) traffic at the application layer. Among various implementation strategies, ModSecurity deployed in conjunction with the OWASP CRS represents the prevailing industry standard for open-source defense. This architecture employs a rule-based deterministic detection engine that uses regular expressions and pattern matching to intercept prevalent exploitation vectors [12]. Although alternative anomaly-based and hybrid approaches have been proposed to address the rigidity of static signatures [20], [21], rule-based systems remain the dominant operational choice due to their predictability and interpretability.

However, the utility of WAFs as a source of threat intelligence is fundamentally limited by their architectural design. The operational imperative of a WAF engine is interception efficacy rather than taxonomical precision. These systems are optimized to enforce a binary decision: block or allow. This design prioritizes minimizing latency and false negatives over granular threat classification. Consequently, a malicious payload often triggers a cascade of generic, high-level rules (e.g., "Inbound Anomaly Score Exceeded" or "Protocol Violation") rather than vector-specific signatures. This phenomenon results in semantic ambiguity, where audit logs record the fact of a block but do not explicitly categorize the nature of the attack.

This operational constraint renders raw WAF Rule IDs an unreliable proxy for ground-truth attribution. Reliance solely on rule activation introduces significant noise, as adversarial obfuscation may bypass specific descriptive signatures while still triggering fallback mechanisms. To derive actionable intelligence, it is necessary to decouple detection from classification. This requires an independent analysis mechanism capable of examining the intrinsic semantics of the payload itself, irrespective of the specific WAF rule triggered, to accurately determine the attack category. This precise classification is a prerequisite for contextualizing incidents and mapping them to standardized higher-level frameworks.

Ultimately, the misalignment between low-level blocking events and high-level strategic intelligence creates a semantic gap. Although WAFs generate voluminous syntax-driven telemetry, they lack the contextual depth required for automated threat hunting. Bridging this gap requires a framework that transforms raw, noisy audit logs into structured, TTP-aligned threat descriptions, enabling security operations to transition from reactive blocking to proactive threat characterization.

B. CTI

CTI encapsulates actionable knowledge about adversaries, attack behaviors, and threat campaigns, allowing defenders to interpret, contextualize, and respond to security incidents

more effectively [22]. CTI is commonly derived from heterogeneous sources, including open-source intelligence (OSINT), incident response records, malware analyses, intrusion reports, and honeypot observations. To support interoperability and operational readiness, CTI is increasingly represented using standardized formats such as STIX 2.1 [23], MITRE ATT&CK [24], and CAPEC [25].

The MITRE ATT&CK framework has emerged as the dominant structure to describe adversarial behavior, organizing more than 700 techniques across 14 tactics, such as Initial Access, Execution, Privilege Escalation, Discovery, and Exfiltration. Its structured representation enables security operations centers (SOCs) to perform behavior-based detection, threat hunting, and adversary emulation using a coherent, cross-platform vocabulary [26]. Automatically mapping operational events to ATT&CK TTPs enhances response workflows by revealing the attacker's position within the kill chain and highlighting which defensive actions should be prioritized [27]. However, a substantial representational mismatch exists between CTI narratives and WAF log data, whose low-level HTTP artifacts lack the semantic and behavioral context required for direct ATT&CK attribution.

This mismatch arises from two fundamental differences in representational granularity. First, CTI provides high-level semantic descriptions of adversarial behavior (e.g., "Exfiltration over Application Protocols"), typically emphasizing attacker intent, procedural steps, and the broader operational objectives behind an intrusion [28]. Second, WAF logs contain low-level syntax-oriented artifacts derived from HTTP request processing, including fragmented URI paths, query parameters, encoded payloads, header fields, and rule-trigger metadata. These entries preserve only the surface-level request syntax (e.g., "id= 1' OR 1 = 1-"), lacking the abstraction necessary to infer intent, execution flow, or placement within a broader attack lifecycle.

Although prior work has explored ATT&CK mapping using endpoint telemetry, malware samples, or system call traces, little research addresses the challenge of deriving TTP-level semantics directly from raw WAF audit logs, which are significantly noisier, less structured, and more ambiguous than other forms of security data. Consequently, a critical opportunity arises: enabling WAF logs, traditionally limited to rule-based detection, to be transformed into CTI-aligned, semantically rich threat intelligence. This motivates the design of our proposed framework, which leverages contrastive learning, structure-aware payload generation, and CTI retrieval to bridge the semantic divide between application-layer logs and high-level threat behavior representations.

III. RELATED WORK

A. Evolution of Web Attack Detection

As discussed in Section II, the primary limitation of contemporary WAFs is their reliance on binary decision logic, which lacks the taxonomical precision required for downstream threat intelligence. Prior research in web attack detection has predominantly focused on identifying malicious intent from Uniform Resource Locators (URLs) or full HTTP requests using supervised learning techniques.

TABLE I
COMPARISON OF EXISTING METHODS FOR ATTACK TYPE CLASSIFICATION FROM WEB LOGS

Paper	Year	Input	Output	Method	Tokenizer	Datasets
[31]	2018	HTTP Request	Malicious/Benign	SVM / Naive Bayes	N-grams	CSIC 2010
[32]	2021	HTTP Request	Malicious/Benign	HTTP2vec (Embedding)	Custom	CSIC 2010, CSE-CIC-IDS2018
[29]	2023	URL	Malicious/Benign	XGBoost	CountVectorizer	ISCX-URL2016
[30]	2024	URL	Malicious/Benign	TF-IDF + Random Forest	CountVectorizer	Kaggle Malicious URLs
[33]	2024	URL	Malicious/Benign	BERT (pre-train) + SSCL	BERT	GLUE, Malicious URLs
[34]	2024	URL	Malicious/Benign	BERT (pre-train)	BERT	Mendeley Web Dataset
Ours	2025	WAF Logs	Attack Types (8 classes)	CodeBERT-CL + K-means	CodeBERT	WAF Logs (MCTS Augmented)

Early approaches focused on feature engineering with traditional classifiers. Nowroozi et al. [29] and Benita et al. [30] employed algorithms such as XGBoost and Random Forest to distinguish malicious URLs. Similarly, Betarte et al. [31] focused on analyzing full HTTP requests, applying Support Vector Machines (SVM) and Multinomial Naive Bayes to detect attacks within the CSIC 2010 dataset. Although effective for known patterns, these methods rely heavily on manual feature extraction and often struggle with the sequential dependencies of obfuscated payloads.

More recent studies have adopted deep learning and embedding paradigms to enhance detection robustness. Gniewkowski et al. [32] proposed HTTP2vec, a representation learning method that embeds HTTP requests into vector spaces. Evaluated on both CSIC 2010 and the large-scale CSE-CIC-IDS2018 dataset, their approach demonstrated that learning semantic representations significantly improves anomaly detection. Following this trend, Li et al. [33] and Mahdaouy et al. [34] leveraged pre-trained BERT models to extract semantic features from URLs, achieving state-of-the-art performance against adversarial perturbations.

However, a critical gap remains: these methods typically formulate the problem as binary classification (Malicious vs. Benign). As summarized in Table I, existing works, including those using comprehensive datasets such as CSIC 2010 and CSE-CIC-IDS2018, typically produce a binary verdict. This binary output mirrors the limitation of operational WAFs; it flags the presence of a threat but fails to categorize its specific nature (e.g., distinguishing SQLi from XSS or LFI).

In contrast, our framework addresses the need for multi-class categorization identified in Section II. Unlike previous works limited to binary outcomes, we utilize the full WAF log context and employ a BERT-based model with contrastive learning. This approach enables the precise identification of eight different attack types, providing the granular labels necessary for the subsequent TTP mapping process.

B. Automated TTP Mapping

Bridging the semantic gap between low-level WAF logs and high-level CTI requires mapping technical artifacts to standardized TTPs (e.g., MITRE ATT&CK). Existing studies have approached this mapping problem using different forms of security evidence, including unstructured CTI reports, structured detection rules, and operational security telemetry.

Early works relied on lexical matching and statistical methods. Husari et al. [35] proposed TTPDrill, which utilizes

BM25 and Part-of-Speech (PoS) tagging to map threat actions from unstructured text to ontology-based representations. Similarly, Legoy et al. [36] applied TF-IDF and SVM to retrieve ATT&CK tactics from CTIs. Although these methods established the foundation for TTP extraction, they are constrained by keyword dependence and often fail to capture the contextual semantics of complex attack descriptions.

Recent advances have integrated LLMs to improve semantic understanding. Rani et al. [37] and Alam et al. [38] employed BERT-based architectures and span-based matching to extract actionable intelligence and attack patterns from CTI narratives with greater precision. Huang et al. [39] proposed MitreTRIEVAL, which retrieves MITRE ATT&CK techniques from unstructured CTI reports by fusing deep learning with ontology-based reasoning. Li et al. [27] further advanced this domain by proposing an automated framework for discovering and mapping tactics from unstructured threat intelligence. These approaches benefit from the relatively rich behavioral descriptions available in CTI narratives, whereas detection rules and machine-generated telemetry generally provide more constrained or fragmented semantic evidence.

ATT&CK mappings have also been derived from structured detection rules. Lai et al. [40] associated Snort IDS rules with TTPs using K-means clustering. Daniel et al. [41] compared machine-learning and LLM-based approaches for assigning ATT&CK labels to 973 Snort rules. Their evaluation shows that LLMs can provide scalable and explainable mappings, although conventional supervised models achieve higher overall labeling accuracy and rare techniques remain difficult to identify reliably. Lee et al. [42] further introduced an LLM-based multi-label approach that converts Snort rules into descriptive representations and maps them to ATT&CK using retrieval-augmented generation. Wudali et al. [43] extended LLM-based attribution to structured SIEM rules and showed that rule content alone may provide insufficient semantic information, making additional contextual knowledge important for accurate mapping. Kurniawan et al. [44] integrated Sigma detection rules with an ATT&CK knowledge graph by translating detection logic into executable queries and linking detected alerts to ATT&CK tactics and techniques. Because these approaches operate primarily on reusable detection-rule semantics, the resulting attribution may not reflect behavioral differences among individual events that trigger the same rule.

ATT&CK attribution has also been explored directly from operational telemetry. Hans et al. [45] used LLMs to analyze sequences of Suricata IDS logs and associate observed network

TABLE II
COMPARISON OF EXISTING TTP RETRIEVAL AND ALIGNMENT APPROACHES

Paper	Year	Input Modality	Output	Feature Representation	Alignment Strategy	Data Augmentation
[35]	2017	CTI	TTPs	Lexical (BM25 + PoS)	Ontology Mapping	No
[36]	2020	CTI	TTPs	Statistical (TF-IDF)	Supervised Classification (SVM)	No
[44]	2021	Sigma Rules + Logs	TTPs	Knowledge Graph	Rule Translation + ATT&CK Linking	No
[37]	2023	CTI	TTPs	Contextual (BERT)	Supervised Classification	No
[38]	2023	CTI	TTPs	Contextual (Sent-BERT)	Span-based Matching	No
[40]	2023	Snort Rules	TTPs	Statistical	Unsupervised Clustering (K-means)	No
[27]	2024	CTI	TTPs	Contextual (DistilBERT)	Supervised Mapping	No
[41]	2025	Snort Rules	TTPs	Contextual (LLM / ML)	Multi-label Mapping	No
[42]	2025	Snort Rules	TTPs	Contextual (LLM)	RAG-based Mapping	No
[43]	2025	SIEM Rules	TTPs	Contextual (LLM)	Multi-stage LLM Mapping	No
[45]	2025	Suricata IDS Logs	TTPs	Contextual (LLM)	LLM-based Mapping	No
Ours	2025	WAF Logs + CTI	TTPs	Augmented Semantics	Embedding Similarity Retrieval	Yes

behaviors with ATT&CK techniques. This approach incorporates runtime evidence beyond static rule definitions, but its semantic view is constrained by the behaviors observable through network-level IDS telemetry and relies on aggregating log sequences into higher-level behavioral phases.

Our work fundamentally differs from prior research in three critical dimensions. First, while existing detection methods produce binary verdicts (Table I), we achieve fine-grained multi-class categorization across eight attack types, providing the taxonomical precision essential for TTP attribution. Second, while prior ATT&CK mapping studies have considered CTI reports, structured detection rules, and other forms of operational security telemetry (Table II), we focus specifically on deriving TTPs from raw WAF audit logs that combine event-specific HTTP and payload content with WAF rule metadata. Third, we integrate structure-aware data generation, contrastive semantic learning, and CTI-grounded retrieval into a unified end-to-end framework. This combination enables WAFs to transition from simple blocking devices into sources of ATT&CK-aligned threat intelligence. To our knowledge, no prior work addresses all three challenges simultaneously in the WAF setting: generating validated diverse attack data, performing fine-grained semantic classification of WAF events, and deriving event-level TTP attributions through CTI-grounded retrieval within a single coherent system.

IV. PROBLEM STATEMENT

The representational gap between low-level WAF logs and high-level cyber threat intelligence creates a fundamental barrier to deriving adversarial intent from raw application-layer logs. WAF audit entries primarily capture syntactic artifacts originating from HTTP and HTTPS request processing, including fragmented URI paths, URL-encoded payloads, parameter sequences, and coarse-grained rule trigger information. In contrast, CTI sources such as MITRE ATT&CK provide semantically structured descriptions of TTPs that articulate the objectives, behavioral patterns, and operational context of an intrusion. Without an intermediate semantic layer connecting these two modalities, WAF alerts remain confined to reactive decision-making and cannot support behavior-driven threat analysis.

This study formulates the main research problem as the task of inferring the most plausible MITRE ATT&CK technique from a given WAF log. Formally, the input to this problem is a raw WAF audit record that contains the syntactic structure of an inbound request and associated rule-trigger metadata. The desired output is one or more ATT&CK techniques that accurately characterize the adversarial behavior underlying the request. The objective is to enrich these low-level artifacts with additional semantic cues derived from CTI repositories to maximize the fidelity and reliability of technique-level attribution. To ensure evidence-level traceability, each attribution is associated with the observable WAF artifacts, the predicted attack type, the LLM-expanded behavioral description, and the retrieved CTI records that support the final ATT&CK mapping.

To illustrate, consider a WAF log entry containing the payload:

```
GET /search.php?id=1' UNION SELECT
username,password FROM users
accompanied by ModSecurity alert tags such as
OWASP CRS /WEB_ATTACK /SQL_INJECTION.
```

The first subproblem should classify this as SQLi based on the syntactic structure and semantic patterns. The second subproblem should then map it to relevant ATT&CK techniques such as T1190 (Exploit Public-Facing Application) and T1552.001 (Unsecured Credentials: Credentials In Files), ranked by semantic similarity to supporting CTI evidence describing SQL injection attacks targeting credential stores.

To make this problem tractable, we decompose it into two subproblems, each addressing a distinct semantic gap in the attribution pipeline.

A. Fine-Grained Attack Classification

The first subproblem concerns the identification of attack types directly from WAF telemetry. Given a raw WAF log as input, the goal is to classify it into one of eight traffic classes: SQLi, XSS, CI, MFU, CSRF, LFI, BF, or Benign traffic. The classification task covers seven malicious classes, while Benign traffic serves as the negative class. The selected attack classes were chosen based on two practical criteria. First, the attack must expose discriminative evidence at the WAF boundary through the HTTP request or associated WAF metadata.

Second, its ground truth must be reproducible in our controlled environment through execution against a corresponding vulnerable endpoint. The objective is to learn invariant semantic representations that are robust to common forms of payload mutation, obfuscation, and structural variation. The constraint is that the analysis must rely solely on features observable at the WAF boundary without access to server-side execution traces or application-layer logic.

B. CTI-Grounded Semantic TTP Attribution

The second subproblem concerns technique extraction through CTI-grounded semantic expansion. The input consists of the WAF log combined with its predicted attack type from the first subproblem. The goal is to produce a behaviorally meaningful description of the event and retrieve the most relevant ATT&CK techniques from a large corpus of CTI reports. The objective is to use a large language model to generate coherent semantic representations of the log that preserve the operational intent of the attack, followed by embedding-based similarity search to identify the techniques most consistent with CTI evidence. The constraint is that the LLM must not introduce speculative or unsupported content; all expansions must remain grounded in observable WAF evidence and the predicted attack type, while the final attribution must be supported by retrieved CTI evidence.

C. Evaluation Criteria

The proposed framework is considered successful if it satisfies the following criteria:

- 1) **Classification Performance:** Achieve strong multi-class classification performance across the eight attack categories, demonstrating robust discrimination between attack types despite payload mutations and obfuscation techniques.
- 2) **Retrieval Quality:** Achieve effective top- k retrieval performance for TTP attribution, such that the correct ATT&CK technique appears within a compact set of ranked candidates.
- 3) **Operational Coverage:** Identify a substantially larger set of unique ATT&CK techniques compared with rule-semantics-based attribution baselines, demonstrating enhanced visibility into adversarial behaviors.
- 4) **Evidence-Level Traceability:** Provide traceable mappings between WAF evidence, predicted attack types, and assigned ATT&CK techniques, supported by retrieved CTI evidence.

Together, these two subproblems and their associated evaluation criteria define an end-to-end semantic attribution task in which raw, syntax-oriented WAF logs are transformed into structured, CTI-aligned representations of adversarial behavior. This formulation provides the conceptual foundation for the framework developed in this work.

V. SOLUTION APPROACH

To address the two subproblems defined in Section IV, we propose a unified framework composed of three cascaded

modules, as illustrated in Figure 1. The framework operates in two distinct phases: a training phase that prepares the classification and retrieval models, and an inference phase that processes live WAF logs to produce TTP attributions.

During the *training phase*, we first employ *Adversarial Payload Generation* to synthesize a diverse dataset of validated attack variants, addressing the data scarcity challenge identified in Section III. This module produces 175,509 payload samples and semantic-verification pairs across eight traffic categories. The attack-classification payloads are then used to train the *BERT-based Defensive Detection* module, which addresses Subproblem 1 by learning robust semantic representations for fine-grained attack classification.

During the *inference phase*, the trained classifier processes incoming WAF logs to identify attack types. The classified logs are then passed to the *CTI-based TTP Retrieval* module, which addresses Subproblem 2 by expanding the log semantics using a large language model and retrieving the most relevant ATT&CK techniques through embedding-based similarity search against a comprehensive CTI database containing 18,088 technique descriptions.

The following subsections detail each module's methodology and operational principles.

A. Adversarial Payload Generation

This module addresses a fundamental challenge in training robust WAF log classifiers: the scarcity of diverse, validated attack data. Relying solely on static public datasets limits model generalization, as real-world attacks exhibit significant syntactic variation through obfuscation, encoding, and evasion techniques. To overcome this limitation, we employ an automated generation mechanism that synthesizes novel attack variants while maintaining semantic validity.

Our approach builds upon the work of Qu et al. [15], which demonstrated the efficacy of MCTS in bypassing WAFs via SQL injection mutations. However, we significantly extend this methodology to address broader operational requirements. Although the original framework focused primarily on a single attack vector, we generalize the state-space search to support eight distinct traffic classes. Furthermore, to overcome the risk of generating syntactically invalid attacks, a limitation often overlooked in purely stochastic mutation, we integrate a tightly coupled Payload Semantic Verification module.

Functionally, this phase operates as a closed-loop system driven by MCTS Payload Generation. The mechanism systematically navigates the high-dimensional mutation space to identify payloads that evade WAF detection. Crucially, to ensure operational utility, the Payload Semantic Verification module acts as a logical gatekeeper. A mutation that bypasses a WAF but loses its executable logic (e.g., broken SQL syntax) is discarded. This module validates whether the mutated payload preserves the original attack semantics. Only payloads that successfully pass this verification are output as Mutated Payloads. These validated samples are then injected into the environment to trigger the WAF, generating the raw logs required for the subsequent training phase.

1) *MCTS Payload Generation:* To effectively explore the high-dimensional space of adversarial examples, we structure

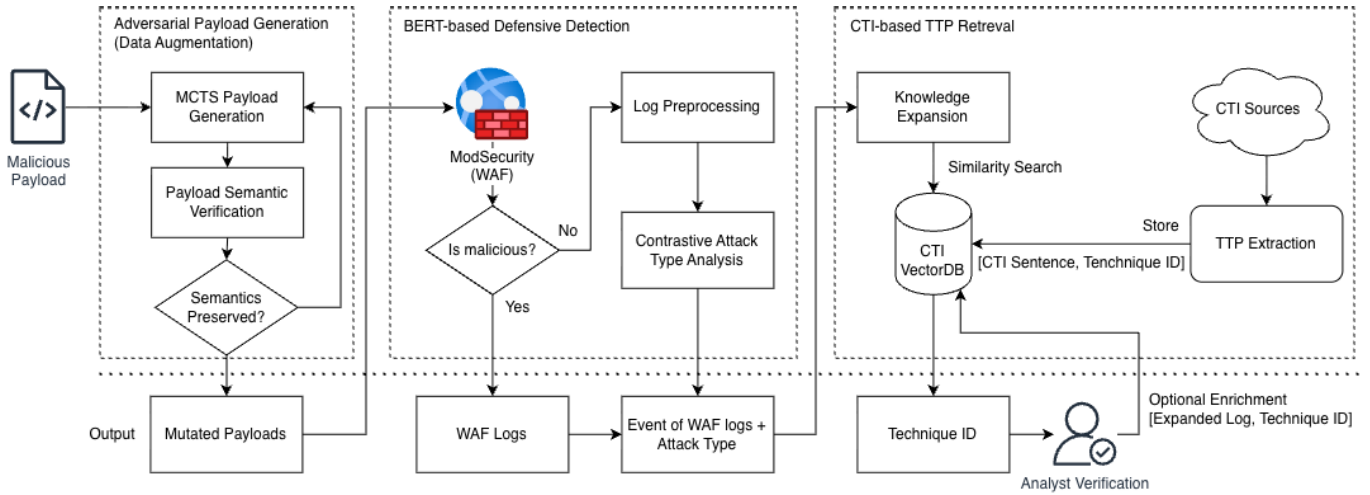


Fig. 1. Overview of the proposed framework.

TABLE III

DEFINITION OF NODE TYPES IN THE HIERARCHICAL TREE REPRESENTATION. OPERABLE NODES SERVE AS CANDIDATES FOR MUTATION ACTIONS

Node Label	Operable	Description
KEYWORD	Yes	Syntax keywords (e.g., OR, AND, SELECT)
WHITESPACE	Yes	Spacing characters (Space, Tab, Newline)
COMMENT	Yes	Comment blocks (e.g., /*...*/, #)
INTEGER	Yes	Numeric values (e.g., 1, 0x1)
OPERATOR	Yes	Logical/Comparison operators (=, <>, LIKE)
TAUTOLOGY	Yes	Boolean tautologies (e.g., 1 = 1)
LOGICAL_INV	Yes	Logical invariants appended to queries
INLINE_CMT	Yes	Comments injected within tokens
WHERE	Yes	Conditional clauses
DML	Yes	Data Manipulation Language statements
IDENTIFIER	No	Table or Column names (e.g., user, passwd)
STRING	No	String literals (e.g., 'admin')

the generation process into three stages: Hierarchical Tree Representation, Context-Free Grammar (CFG) Mutation, and MCTS optimization.

a) *Hierarchical tree representation*: Directly mutating raw strings often leads to broken syntax. To maintain structural integrity, we parse each payload into a hierarchical tree (Abstract Syntax Tree) based on the HTTP protocol and attack-specific grammars. As defined in Table III, we classify tree nodes into Operable and Inoperable types. Operable nodes (e.g., Keywords, Operators) serve as potential anchor points for mutation actions, while Inoperable nodes (e.g., Identifiers) remain fixed to preserve the target logic.

b) *Mutation with CFG*: We define a set of mutation operators formally described by a Context-Free Grammar. Rather than relying on a single source, we synthesized a comprehensive library of eleven fundamental mutation strategies by consolidating distinct adversarial techniques from existing literature. These strategies include Case Swapping, Whitespace Substitution, Comment Injection, Comment Rewriting, Integer Encoding, Operator Swapping, Logical Invariant, Inline Comment, Where Rewriting, DML Substitution, and Tautology Substitution. To ensure robustness across various attack vectors, we adapted these operators by analyzing

TABLE IV

MUTATION APPLICABILITY MATRIX ACROSS EIGHT ATTACK CLASSES: SQL INJECTION (SQLi), CROSS-SITE SCRIPTING (XSS), COMMAND INJECTION (CI), MALICIOUS FILE UPLOAD (MFU), LOCAL FILE INCLUSION (LFI), BRUTE FORCE (BF), CROSS-SITE REQUEST FORGERY (CSRF), AND BENIGN TRAFFIC. (✓: FULLY APPLICABLE, △: PARTIALLY APPLICABLE, X: INAPPLICABLE)

Mutation Operator	SQLi	XSS	CI	MFU	LFI	BF	CSRF	Benign
Case Swapping	✓	✓	✓	✓	✓	X	X	-
Whitespace Substitution	✓	✓	✓	✓	✓	X	X	-
Comment Injection	✓	△	✓	X	✓	X	X	-
Comment Rewriting	✓	△	△	X	✓	X	X	-
Integer Encoding	✓	X	X	X	✓	X	X	-
Operator Swapping	✓	△	△	X	X	X	X	-
Logical Invariant	✓	△	✓	X	X	X	X	-
Inline Comment	✓	X	X	X	✓	X	X	-
Where Rewriting	✓	X	X	X	X	X	X	-
DML Substitution	✓	△	△	X	X	X	X	-
Tautology Substitution	✓	✓	✓	X	△	X	X	-

domain-specific research. Specifically, we incorporated SQL injection automation frameworks [46], [47], [48], XSS obfuscation techniques [15], [49], and command injection syntax [50]. Additionally, structural mutations applicable to broader traffic classes were generalized from related work [51], [52], [53].

Table IV presents the Mutation Applicability Matrix, detailing how these consolidated operators are assigned to our eight target traffic classes. The symbols denote the degree of syntactic compatibility: fully applicable (✓), conditionally applicable (△), and structurally incompatible (X). This distinction is critical because different interpreters process the syntax differently. For example, *Logical Invariant* (e.g., appending AND 1=1) is fully effective for SQLi but structurally incompatible (X) for attacks such as LFI, where the input is interpreted as a file path rather than a logic statement. In contrast, the triangle (△) indicates techniques that are valid only in specific contexts. A prime example is *Comment Injection*: while SQL engines allow comments (e.g., /**/) to replace spaces arbitrarily,

XSS payloads are constrained by HTML parser rules. Injecting comments inside an HTML tag often breaks the syntax unless placed within specific JavaScript contexts, hence it is marked as partially applicable.

Appendix provides representative mutation examples for the attack-specific payload classes used in our MCTS-based generation process. These examples illustrate how the mutation operators in Table IV are instantiated for SQLi, XSS, CI, LFI, and MFU payloads. BF and CSRF are not listed because BF is generated using dictionary-based login attempts, while CSRF is represented through request-flow templates rather than CFG-based payload mutation.

c) *MCTS Optimization*: To navigate the vast mutation space efficiently, we model the payload generation as a Markov Decision Process (MDP) and solve it using MCTS. Unlike random fuzzing, MCTS is a heuristic-driven search algorithm that combines tree search with reinforcement learning principles. The search process iterates through four distinct phases (Selection, Expansion, Simulation, and Backpropagation) to progressively converge on payloads that successfully bypass the WAF.

The process begins with the *Selection* phase, where the algorithm traverses the tree from the root to select the most promising node for mutation. To balance the trade-off between exploiting known high-reward branches and exploring unvisited mutations, we employ the Upper Confidence Bound for Trees (UCT) policy. The node i is selected to maximize the following value:

$$UCT_i = \frac{w_i}{n_i} + C \cdot \sqrt{\frac{\ln N}{n_i}} \quad (1)$$

where w_i is the accumulated reward of node i , n_i is the visit count of node i , N is the total visit count of the parent node, and C is the exploration constant.

Once a leaf node is reached, the *Expansion* phase applies a mutation operator from our CFG to create a new child node, representing a novel payload variant. This is immediately followed by the *Simulation* phase, which constitutes the critical WAF interaction loop. The generated tree structure is serialized into a raw HTTP string and transmitted to the target WAF. The environment provides immediate feedback based on the response. A successful bypass (e.g., HTTP 200 OK) yields a maximal reward ($R = 1.0$), while time-based side-channel responses yield a high partial reward ($R = 0.95$). In contrast, blocked requests (HTTP 403) receive $R = 0.0$. SQL syntax errors observed during WAF testing receive a small non-zero reward ($R = 0.05$), indicating that the payload reached the backend interpreter but did not result in a successful bypass. Rendering or request-level failures are handled using a fallback length-penalized reward sampled from $[0, 0.5]$.

Finally, in the *Backpropagation* phase, the reward obtained R is propagated through the tree, updating the values of w_i and n_i for all ancestor nodes, thus influencing future selection decisions.

It is crucial to note that while this optimization process maximizes the probability of WAF evasion, it does not guarantee the functional correctness of the generated code. MCTS may discover mutations that bypass the WAF simply because they

are structurally broken, rendering them operationally useless. Consequently, the output of the MCTS engine is treated solely as a set of candidate payloads. These candidates are not stored immediately but instead are forwarded to the subsequent Payload Semantic Verification module for rigorous functional validation.

2) *Payload Semantic Verification*: A critical challenge in MCTS-based payload generation is ensuring that mutated payloads preserve the original adversarial semantics. To address this, we train a Siamese network using DistilBERT [54] as the encoder to learn a semantic embedding space where functionally equivalent payloads are mapped to nearby points.

We construct a training dataset of 92,000 payload pairs, where each pair consists of an original attack and its mutated variant. Pairs are labeled as either *semantically preserved* (the mutation maintains attack functionality) or *semantically divergent* (the mutation breaks the attack logic). The Siamese architecture processes both payloads through a shared DistilBERT encoder, producing two embeddings $\mathbf{e}_{\text{orig}}$ and $\mathbf{e}_{\text{mut}}$. We then compute the cosine distance between these embeddings:

$$d(\mathbf{e}_{\text{orig}}, \mathbf{e}_{\text{mut}}) = 1 - \frac{\mathbf{e}_{\text{orig}} \cdot \mathbf{e}_{\text{mut}}}{\|\mathbf{e}_{\text{orig}}\| \|\mathbf{e}_{\text{mut}}\|} \quad (2)$$

The model is trained using the margin-based contrastive loss proposed by Hadsell et al. [55], with margin m :

$$\mathcal{L}_{\text{contrastive}} = y \cdot d^2 + (1 - y) \cdot \max(0, m - d)^2 \quad (3)$$

where $y = 1$ for semantically preserved pairs and $y = 0$ for divergent pairs. This loss function pulls similar payloads together in embedding space while pushing dissimilar ones apart by at least margin m .

During the MCTS generation phase, this pre-trained verification model acts as a semantic gatekeeper. For every candidate payload generated, we encode both the candidate and its parent payload using the trained DistilBERT encoder, then calculate the cosine distance d between their embeddings. Only candidates with distance d below a strict threshold $\tau = 0.48$ are deemed to have preserved the adversarial intent and accepted into the dataset. The threshold $\tau = 0.48$ was selected based on validation-set performance and then fixed for subsequent payload filtering. This approach effectively filters out semantically incoherent mutations without requiring risky and slow dynamic execution in a sandbox environment.

B. BERT-Based Defensive Detection

This module addresses Subproblem 1 (Fine-Grained Attack Classification) defined in Section IV by transforming raw WAF logs into precise attack category labels. Unlike traditional WAFs that rely on rigid regex matching, our approach utilizes CodeBERT, a pre-trained language model specialized for programming languages, to capture the structural and behavioral patterns of web attacks [56].

The module employs a two-stage architecture: supervised contrastive learning to create discriminative embeddings, followed by K-means clustering to enable both supervised classification of known attack types and unsupervised discovery of novel variants. This hybrid approach ensures robustness against obfuscation while maintaining the flexibility to detect

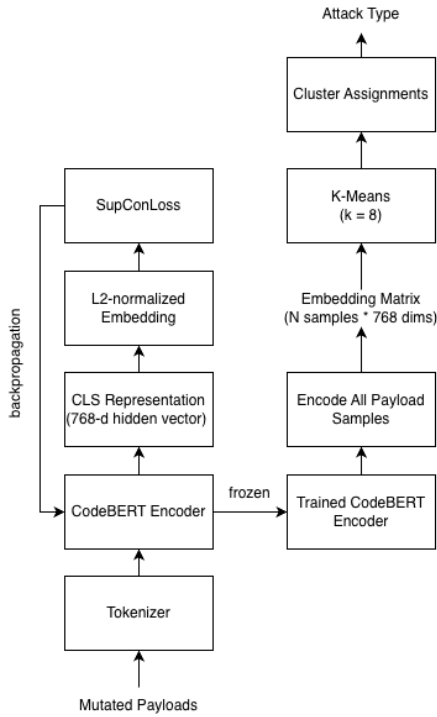


Fig. 2. Architecture of the attack type analyzer.

previously unseen attack patterns. The module comprises two key components: Log Preprocessing with Structure-Aware Tokenization and Contrastive Attack Type Analysis.

1) Log Preprocessing: Raw WAF logs contain substantial noise that can hinder model generalization. To mitigate this, we employ a preprocessing pipeline designed to remove environment-specific artifacts while preserving the syntactic structure of the attack. First, we apply token masking to replace dynamic fields such as IP addresses, timestamps, session identifiers, and port numbers with generic placeholders (e.g., [IP], [PORT]), preventing the model from overfitting to deployment-specific features. URL-encoded characters (e.g., %27 → ') are also decoded to expose the underlying attack syntax.

Following preprocessing, the choice of tokenizer plays a critical role in feature extraction. Standard natural-language tokenizers, such as BERT's WordPiece, are optimized for human language and often fragment technical artifacts into semantically meaningless sub-tokens. For example, a filename such as `WerFault.exe` is typically split into `['Wer', '##Fault', '##.', '##exe']`, disrupting the structural unity of the token. To address this issue, we adopt CodeBERT's tokenizer, which is trained on large-scale code corpora containing text rich in symbols and non-natural language (e.g., API names, file paths, operators, string literals). As a result, it produces more stable and coherent subtokenization for payloads that mix code-like fragments with URL parameters. This better preserves the operational structure of the request, enabling the downstream model to capture the syntactic and semantic properties of diverse attack payloads.

2) Contrastive Attack Type Analysis: Following tokenization, the processed sequence is fed into the Attack Type Analyzer to classify the log into one of eight specific traffic classes: SQLi, XSS, CI, MFU, LFI, BF, CSRF, and Benign. As illustrated in Figure 2, the workflow is structured into two sequential stages: Supervised Contrastive Fine-Tuning and Cluster-based Classification.

In the initial fine-tuning stage, we utilize the codebert-base model as the backbone encoder. The input sequence is encoded into a 768-dimensional hidden vector, where the representation of the [CLS] token is extracted and L2-normalized to produce the projection vector z . To shape a discriminative embedding space, the model is optimized using the supervised contrastive loss (L_{SupCon}) proposed by Khosla et al. [57]. Unlike standard Cross-Entropy, L_{SupCon} leverages label information to pull embeddings of the same class together while pushing apart distinct classes. This formulation is suitable for attack-type representation learning because each anchor can leverage multiple same-class samples as positive examples, encouraging compact within-class representations and greater separation between different attack classes. Formally, for a batch of N samples, the loss is defined as:

$$L_{SupCon} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{a \in A(i)} \exp(z_i \cdot z_a / \tau)} \quad (4)$$

where i is the index of the anchor sample, $P(i)$ is the set of positive samples (same class as i) in the batch, $A(i)$ includes all samples except i , and τ is the temperature parameter that controls the smoothness of the distribution. This optimization forces the model to learn a geometric structure in which semantically similar attacks form compact clusters.

Upon completion of training, the fine-tuned CodeBERT encoder is frozen and utilized as a static feature extractor. We encode all payload samples to generate a comprehensive embedding matrix of size $N \times 768$. Finally, to assign specific attack labels, we apply K-Means Clustering with $k = 8$. The algorithm partitions the embedding space by minimizing the within-cluster sum of squares (WCSS). The objective function J is given by:

$$J = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} \|\mathbf{x} - \boldsymbol{\mu}_j\|^2 \quad (5)$$

where C_j represents the set of samples assigned to the j -th cluster, and $\boldsymbol{\mu}_j$ is the centroid of cluster C_j . The final prediction for a given log is determined by the cluster C_j to which its embedding $\mathbf{x}$ is assigned. This decoupled approach ensures that classification boundaries are determined by the high-dimensional manifold structure learned by CodeBERT, allowing for the detection of variant patterns that reside within valid cluster boundaries.

C. CTI-Based TTP Retrieval

This module addresses Subproblem 2 (CTI-Grounded Semantic TTP Attribution) defined in Section IV by bridging the semantic gap between low-level technical artifacts (WAF logs with predicted attack types) and high-level adversarial

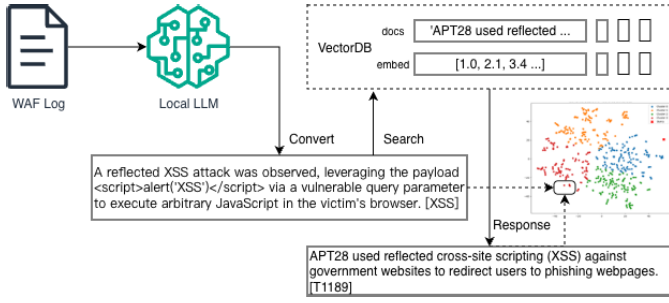


Fig. 3. Workflow of CTI-based TTP retrieval.

behaviors represented by MITRE ATT&CK techniques. The core challenge is that WAF logs are composed of rigid machine syntax, while threat intelligence is written in unstructured natural language. Direct keyword search fails because the syntactic features in logs (e.g., ' OR 1=1--') bear little lexical similarity to CTI descriptions (e.g., "SQL injection targeting authentication bypass").

To overcome this representational mismatch, we implement a three-stage Retrieval-Augmented Generation (RAG) workflow consisting of CTI Database Construction, Knowledge Expansion, and Semantic Similarity Search. The retrieval process is illustrated in Figure 3.

1) *CTI Database Construction*: To establish a comprehensive and searchable knowledge base, we leveraged the automated TTP extraction system proposed by Li et al. [27]. We processed a large corpus of 18,088 unstructured CTI records to extract mapping relationships between attack descriptions and specific MITRE ATT&CK techniques. This extensive dataset provides broad coverage of the threat landscape, encompassing 216 distinct ATT&CK techniques and 475 sub-techniques. Each extracted record is converted into a high-dimensional vector using an embedding model and stored in ChromaDB, a high-performance vector database. This pre-built repository serves as the retrieval corpus, where each CTI record retains its associated ATT&CK Technique ID for subsequent TTP attribution.

2) *Knowledge Expansion via Local LLM*: Raw WAF logs lack the narrative context required for effective matching against CTI reports. To address this, we introduce a Knowledge Expansion module that acts as a semantic translator. This module is designed to ingest comprehensive context, including raw WAF logs, ModSecurity alert messages, and OWASP CRS tags. Crucially, for evasive threats where standard rules remain silent, the module integrates the semantic predictions from our proposed classifier to ensure complete visibility.

We use Llama-3.1-8B-Instruct, an 8-billion-parameter model, as the default inference engine for this task. The model is deployed locally using Hugging Face Transformers with FP16 precision, and no integer quantization (e.g., INT8 or INT4) is applied. Greedy decoding is used with `do_sample=False` and `max_new_tokens=200`. A key advantage of our approach is its modularity [58]. Since the expansion task relies on the model's general reasoning capabilities rather than domain-specific memorization, it operates effectively in a zero-shot manner without requiring

task-specific fine-tuning. This allows the underlying LLM to be easily swapped or upgraded (e.g., to proprietary in-house models) without retraining the pipeline. For instance, a raw input containing `<script>alert(1)</script>` is transformed into a descriptive narrative: "A reflected XSS attack was observed, leveraging a script payload via a vulnerable query parameter to execute arbitrary JavaScript in the victim's browser." This reconstruction aligns the semantic context of the log with the CTI records in our database.

3) *Semantic Similarity Search and Dataset Enrichment*: In the final step, the expanded narrative serves as the query for the retrieval engine. We perform a vector similarity search against the ChromaDB backend to identify CTI records that share the closest semantic proximity to our query. The similarity score S between the query vector $\mathbf{q}$ and a document vector $\mathbf{d}$ is calculated using cosine similarity:

$$S(\mathbf{q}, \mathbf{d}) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\| \|\mathbf{d}\|} = \frac{\sum_{i=1}^n q_i d_i}{\sqrt{\sum_{i=1}^n q_i^2} \sqrt{\sum_{i=1}^n d_i^2}} \quad (6)$$

The system retrieves the top- k CTI records with the highest similarity scores. Because each retrieved record is associated with an ATT&CK Technique ID, the retrieval results form a ranked set of candidate techniques for the corresponding WAF log. The Technique ID associated with the top-ranked CTI record is used as the primary attribution.

To allow the system to evolve, we designed a dataset enrichment mechanism. While the retrieval process is automated, the framework supports a feedback loop where matched pairs (Expanded Log, TTP) can be written back to the CTI VectorDB. To ensure the integrity of the knowledge base, this update process requires manual verification by security analysts. This quality assurance step prevents low-fidelity data from polluting the database while allowing the system to progressively expand its coverage of novel attack patterns.

D. End-to-End Pipeline Summary

The complete framework operates as follows. In the training phase, the Adversarial Payload Generation module (Section V-A) synthesizes diverse attack variants, which are executed against a controlled WAF environment to generate labeled training data. This data is used to train the BERT-based Defensive Detection module (Section V-B), producing a classifier capable of identifying eight attack types from raw WAF logs.

In the inference phase, when a new WAF log is encountered, the system executes the following steps:

- 1) The log undergoes preprocessing and tokenization (Section V-B.1)
- 2) The trained CodeBERT-CL encoder generates a semantic embedding
- 3) K-means clustering assigns the log to one of eight attack categories
- 4) The classified log, combined with WAF metadata, is passed to the Knowledge Expansion module (Section V-C.2)
- 5) An LLM transforms the syntax-heavy log into a behaviorally meaningful narrative

- 6) Vector similarity search retrieves the top- k most semantically similar CTI records from ChromaDB
- 7) The ATT&CK technique associated with the highest-ranked record is assigned to the log

This end-to-end pipeline transforms raw WAF telemetry into actionable, ATT&CK-aligned threat intelligence, enabling security analysts to transition from reactive blocking to proactive, behavior-centric threat hunting. The framework's modularity allows individual components to be upgraded independently. For example, the LLM can be swapped for newer models, or the classifier can be retrained on emerging attack patterns, without redesigning the entire system.

VI. IMPLEMENTATION

To validate the proposed framework, we established a comprehensive experimental testbed integrating automated attack generation, real-time WAF defense, and log analysis. This section details the hardware specifications, network topology, and dataset composition.

A. Experimental Setup and Environment

All experiments were conducted on a high-performance server running Ubuntu 20.04 $\times$ 64. The hardware configuration includes an 11th Gen Intel(R) Core(TM) i7-11700 CPU @ 2.50 GHz, 48 GB of RAM, and an NVIDIA RTX A5000 graphics card to accelerate the training of BERT-based models.

To simulate a realistic web defense scenario while ensuring ground-truth validation, we deployed a containerized environment using Docker, as illustrated in Figure 4. The topology consists of three core components connected via a private Docker Bridge Network:

- **Target Application (DVWA):** A Damn Vulnerable Web Application instance serving as the victim target, listening on port 8080. DVWA provides intentionally vulnerable endpoints that enable controlled exploitation and verification of attack success.
- **WAF (ModSecurity):** Configured as a reverse proxy listening on port 8888, intercepting all incoming HTTP traffic destined for the DVWA. The WAF is configured with OWASP CRS v3.3 to provide baseline protection and comprehensive log generation.
- **Database (MySQL):** A backend database supporting the DVWA, listening on port 3306.

During data collection, the physical PC acts as the attacker, injecting generated payloads into the ModSecurity container via HTTP requests. Each payload is executed against the actual DVWA application, and the resulting audit logs are extracted via a mounted volume for subsequent processing. This setup ensures that our dataset consists of genuine WAF logs generated from actual HTTP transactions rather than synthetic text simulations.

B. Dataset Preparation

The quality and diversity of the training data are critical for the three-module framework described in Section V. We constructed four complementary datasets, each serving a specific role in training or evaluating different components of the pipeline. Table V summarizes the dataset statistics.

TABLE V
DETAILED STATISTICS OF THE DATASETS USED IN THIS STUDY. LLM EXPANSION DENOTES LLM-BASED KNOWLEDGE EXPANSION FOR RETRIEVAL QUERY CONSTRUCTION

Category	Metadata (Class/Type)	Train	Valid	Test
Payloads	XSS	26,605	3,326	3,325
	SQLi	12,320	1,540	1,540
	CI	5,330	666	666
	MFU	9,821	1,228	1,228
	CSRF	6,868	858	859
	Benign	2,115	264	265
	BF	1,603	201	200
	LFI	2,145	268	268
<i>Total Payloads</i>		<i>66,807</i>	<i>8,351</i>	<i>8,351</i>
Verification	Payload & Mutated Pairs	73,600	9,200	9,200
CTIs	CTI & TTP Pairs	18,088 (no split)		
WAF Logs	Log-TTP pairs (w/o LLM expansion)	-	-	52,045
	Log-TTP pairs (w/ LLM expansion)	-	-	9,008

1) *Payload Dataset:* Seed payloads were collected from public benchmarks (DeepXSS [59] and HttpParamsDataset [60]) and Kali Linux penetration-testing utilities. The test seeds were separated before MCTS augmentation, and variants generated from these held-out seeds were retained only in the test set and excluded from the training and validation sets. This prevents overlap between the training and testing seed lineages while preserving mutated test samples for evaluating generalization and robustness to payload variation. MCTS augmentation was selectively applied to under-represented attack types. The resulting dataset contains 83,509 samples across eight traffic classes: XSS, SQLi, CI, MFU, CSRF, Benign, BF, and LFI, including 66,807 training samples, 8,351 validation samples, and 8,351 testing samples.

2) *Semantic Verification Dataset:* This dataset consists of 92,000 payload pairs, each containing an original attack and its mutated variant. Pairs were labeled as semantically preserved or semantically divergent based on whether the mutation maintains attack functionality. The dataset was split into 73,600 training pairs, 9,200 validation pairs, and 9,200 testing pairs for supervising the Payload Semantic Verification module (Section V-A2).

3) *CTI Knowledge Base:* The CTI database comprises 18,088 threat intelligence records paired with ATT&CK technique identifiers, extracted using the automated system proposed by Li et al. [27]. This knowledge base covers 216 distinct ATT&CK techniques and 475 sub-techniques, forming the retrieval index for the CTI-based TTP Retrieval module (Section V-C). This dataset serves as a static retrieval corpus without train/validation/test splits.

4) *WAF Log Dataset:* This dataset contains 61,053 WAF audit log entries for end-to-end TTP attribution evaluation. These logs were generated by executing validated attack payloads, including MCTS-augmented variants, against the ModSecurity/DVWA environment. For retrieval evaluation, the log-TTP pairs are organized into two query-construction settings: one using raw WAF-log inputs directly, and the other using LLM-expanded behavior-level descriptions as retrieval queries. The overall dataset statistics are summarized in Table V. The WAF-log dataset is used only for TTP attribution

TABLE VI
MAPPING BETWEEN EXTERNAL MODSECURITY RULE IDS AND EVALUATION LABELS

Evaluation Label	External Rule IDs	Rule Semantics	Used	Reference ATT&CK Labels
SQLi	942100, 942140, 942160, 942170, 942190, 942240, 942270, 942280, 942360	SQL injection attacks, including libinjection-based SQLi, blind or time-based SQLi, MSSQL-specific patterns, and SQLi-related database keywords.	Yes	T1190
XSS	941110, 941120, 941130, 941140, 941170, 941180	Cross-site scripting attacks, including script tags, event handlers, attribute vectors, JavaScript URI vectors, and blacklist-based XSS detections.	Yes	T1190
CI	932105, 932115, 932130, 932150, 932160, 932170, 934100, 944100, 944110, 944130	Command injection and application-layer RCE attempts, including Unix and Windows command execution, shell expressions, Shellshock, Node.js injection, and Java deserialization or RCE indicators.	Yes	T1059, T1190
LFI	930100, 930110, 930120, 930130	Path traversal, operating-system file access, and restricted file access attempts, including requests for sensitive local files and configuration files.	Yes	T1005, T1190
MFU	933110	PHP script upload detection; treated as the most specific external rule for malicious file-upload-related behavior.	Yes	T1505.003, T1190
CSRF	N/A	No reliable CRS rule group corresponding to CSRF was identified in the external dataset.	No	N/A
BF	N/A	No reliable login brute-force rule group was identified in the external dataset; bot and scanner rules were not treated as BF labels.	No	N/A
Benign	N/A	The external dataset consists of blocked or suspicious ModSecurity audit logs and does not provide benign traffic ground truth.	No	N/A

evaluation and is not included in model training or CTI index construction.

5) *External Real-World ModSecurity Dataset*: To further assess whether the learned representations generalize beyond the payload distributions produced by our MCTS and CFG-based mutation operators, we additionally used an independent real-world ModSecurity dataset for external evaluation. The dataset, released by Lucz and Forstner [19], contains malicious HTTP requests collected over thirty consecutive days, from 27 July 2025 to 25 August 2025, on a production web server protected by OWASP ModSecurity and the OWASP CRS. Unlike the controlled DVWA-based dataset constructed in this study, this external corpus was collected from an uncontrolled operational environment and contains naturally occurring blocked requests, including SQL injection, cross-site scripting, local file inclusion, scanner probes, malformed requests, and evasive input patterns.

We used this dataset exclusively as an external test corpus. It was not used for training the CodeBERT-CL encoder, fitting K-means clusters, training the semantic verification module, selecting decision thresholds, or constructing the CTI retrieval index. Since the dataset provides ModSecurity audit information and triggered CRS rule IDs rather than the attack-type and ATT&CK annotations required for our evaluation, we derived reference labels from sufficiently specific CRS rule semantics. Table VI summarizes the CRS rule groups used to define the external evaluation classes and their associated ATT&CK reference labels. SQLi, XSS, CI, LFI, and MFU were retained because they can be mapped to sufficiently specific CRS rule groups. CSRF, BF, and Benign were excluded from the external evaluation because the dataset does not provide

reliable CRS rule groups or benign ground truth for these categories.

VII. EXPERIMENTAL RESULTS

This section presents a comprehensive evaluation of the proposed framework, assessing its efficacy in semantic verification, attack type classification, and automated TTP retrieval. The experimental design aims to address the following six key research questions (RQs):

- **RQ1 (Semantic Verification)**: How effective is the contrastive semantic verification module in distinguishing between semantically valid mutations and invalid noise?
- **RQ2 (Attack Classification)**: To what extent does the CodeBERT-based analyzer improve multi-class classification performance compared to traditional baselines?
- **RQ3 (TTP Retrieval)**: How effective is the proposed CTI-grounded retrieval module compared with direct TTP attribution baselines, and how do LLM-based knowledge expansion and attack-type context affect retrieval quality?
- **RQ4 (TTP Coverage)**: How does our framework compare to industry-standard rule sets (OWASP CRS) in terms of TTP coverage and granular mapping capabilities?
- **RQ5 (External Generalization)**: Can the proposed classifier and TTP retrieval module generalize to real-world ModSecurity audit logs that were not generated by the adversarial payload generation phase?
- **RQ6 (Runtime and Resource Overhead)**: What are the end-to-end latency, throughput, GPU dependency, and CTI vector database memory overhead of the proposed inference pipeline?

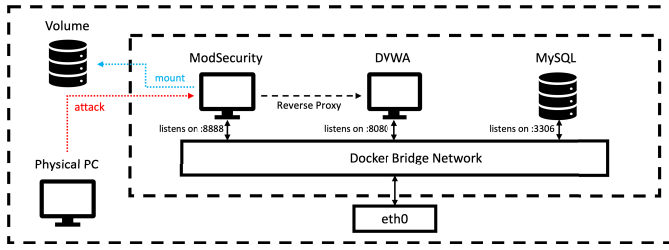


Fig. 4. Network topology of the experimental testbed.

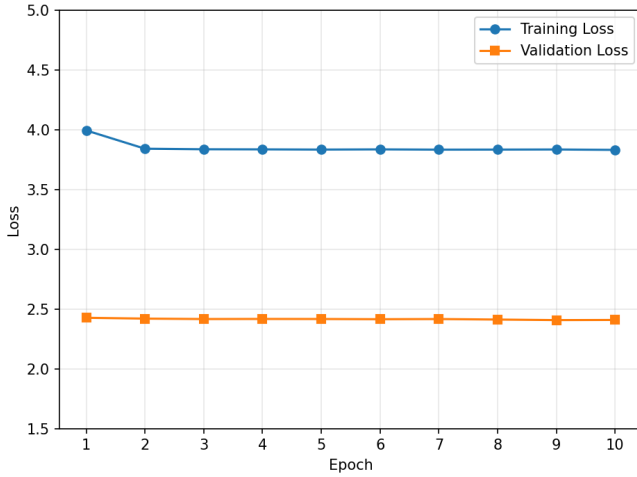


Fig. 5. Training and validation loss of the CodeBERT encoder using supervised contrastive learning.

A. Evaluation of Semantic Verification Effectiveness (RQ1)

The semantic verification module acts as a binary classifier that determines whether an MCTS-generated payload preserves the adversarial semantics of its original seed. A labeled dataset of 92,000 payload pairs was used for training and evaluation, where each pair contains an original attack vector and its mutated variant.

The classifier exhibits near-perfect discriminative ability. In the 9,200-sample testing subset, it achieved an AUC of 0.9991, and with an optimized decision threshold of 0.48, it reached an overall accuracy of 99.98%, with Precision, Recall, and F1-score all equal to 0.9988. The extremely low false-positive and false-negative rates confirm that the module enforces semantic consistency with high reliability and does not admit mutated payloads that deviate from the original attack logic.

To assess the validity of the payload generation process, the module was further applied to an independent set of 5,325 MCTS-generated payloads. Among them, 2,918 (54.80%) were identified as semantically preserved, while 2,407 (45.20%) were labeled semantically divergent. This outcome highlights the inherently stochastic nature of mutation-based payload synthesis and underscores the necessity of the semantic verification step: without it, nearly half of the generated payloads would introduce semantic drift rather than adversarial diversity.

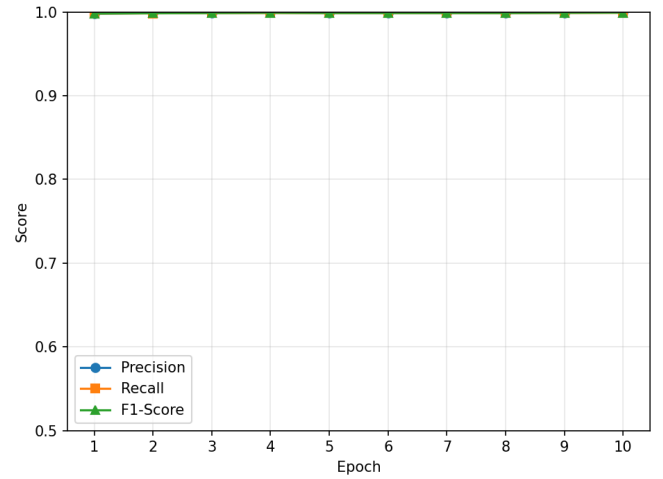


Fig. 6. Precision, Recall, and F1-score curves across training epochs.

TABLE VII
CLASS-WISE PERFORMANCE OF THE CODEBERT-CL ENCODER

Class	Precision	Recall	F1-score	Support
Benign	0.9962	0.9962	0.9962	265
BF	1.0000	0.9950	0.9975	200
CI	0.9985	0.9985	0.9985	666
CSRF	1.0000	1.0000	1.0000	859
LFI	1.0000	1.0000	1.0000	268
SQLi	1.0000	0.9987	0.9994	1540
MFU	1.0000	1.0000	1.0000	1228
XSS	0.9991	1.0000	0.9995	3325
Accuracy			0.9994	8351
Macro Avg	0.9992	0.9986	0.9989	8351
Weighted Avg	0.9994	0.9994	0.9994	8351

B. Performance of Multi-Class Attack Classification (RQ2)

To answer RQ2, we evaluate the system's ability to distinguish between eight distinct traffic classes. The evaluation is conducted in two stages: first, assessing the quality of the representation learned by the CodeBERT encoder (CodeBERT-CL), and second, validating the effectiveness of the clustering-based classification (CodeBERT-CL + K-Means).

1) *Encoder Representation Quality*: We trained the CodeBERT encoder using supervised contrastive loss for 10 epochs. Training convergence trends are illustrated in Figure 5 (training/validation loss) and Figure 6 (Precision/Recall/F1-score).

Table VII details the per-class performance of the encoder. The model achieves an overall accuracy of 99.94% and a macro F1-score of 0.9989. All eight traffic classes achieve F1-scores above 0.996, with CSRF, LFI, and MFU reaching an F1-score of 1.0. The consistently high precision and recall across the evaluated classes indicate that the tokenizer and encoder effectively capture discriminative syntactic and semantic characteristics of different web attack patterns.

2) *Effectiveness of Cluster-Based Classification*: Furthermore, we evaluated the performance of using K-Means

TABLE VIII

PERFORMANCE COMPARISON BETWEEN THE ENCODER (CODEBERT-CL) AND THE CLUSTERING APPROACH (K-MEANS)

Method	Precision	Recall	F1-score
CodeBERT-CL	0.9992	0.9986	0.9989
CodeBERT-CL + K-Means	0.9932	0.9945	0.9938

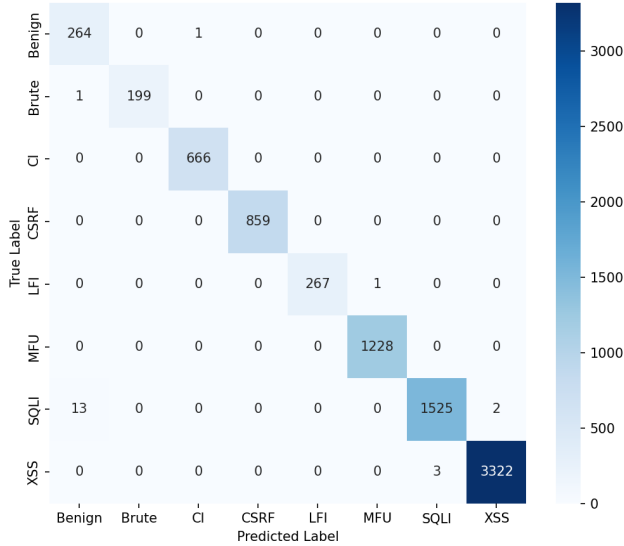


Fig. 7. Confusion matrix (Heatmap) of the K-Means clustering results.

clustering ($k = 8$) on frozen embeddings to assign attack types. Table VIII compares the performance of direct encoder output (CodeBERT-CL) versus the clustering approach (CodeBERT-CL + K-Means).

As observed, introducing K-Means results in only a small performance reduction, with the F1-score decreasing from 0.9989 to 0.9938. This slight reduction is expected because, unlike the encoder, which utilizes supervised label information during representation learning, K-Means assigns samples solely according to the geometric Euclidean distance in the embedding space. The fact that K-Means maintains a high F1-score demonstrates that supervised contrastive learning successfully shapes a highly discriminative embedding space in which samples of the same attack type form compact and well-separated clusters. Figure 7 visualizes this result through the confusion matrix of the K-Means predictions. The strong diagonal dominance further confirms that the learned clusters align closely with the ground-truth attack types.

C. Ablation Study on TTP Retrieval (RQ3)

We investigate the factors that influence the performance of TTP retrieval. We conducted an ablation study to quantify the impact of two key retrieval components: LLM-based knowledge expansion and attack-type context produced by the Contrastive Attack Type Analyzer. Additionally, we evaluated the effectiveness of our retrieval strategy in bridging the semantic gap between WAF logs and CTI records.

TABLE IX

ABLATION STUDY OF LLM-BASED KNOWLEDGE EXPANSION AND ATTACK-TYPE CONTEXT FOR CTI-BASED TTP RETRIEVAL

Exp.	Query Setting	Top@1	Top@3	Top@5	Top@10
A	LLM + attack type	0.5787	0.8635	0.8710	0.8762
B	LLM only	0.0719	0.3102	0.5615	0.6483
C	Raw payload only	0.0777	0.0778	0.0778	0.1108
D	Raw payload + attack type	0.7780	0.7972	0.7996	0.8022

To assess the retrieval quality, we utilize the Top-K Accuracy metric. Given the semantic ambiguity of cyber attacks in which a single payload may correspond to multiple valid techniques, Top-K provides a more holistic evaluation than strict exact matching. The metric is defined as:

$$\text{Top-}K \text{ Accuracy} = \frac{1}{N} \sum_{i=1}^N \mathbb{1} \left(y_i \in \hat{Y}_i^{(K)} \right) \quad (7)$$

where N is the total number of samples, y_i is the ground-truth TTP label for the i -th sample, $\hat{Y}_i^{(K)}$ is the set of predicted K-top labels returned by vector search, and $\mathbb{1}(\cdot)$ is the indicator function.

Table IX presents a component-level ablation of the CTI-based TTP retrieval module. The four settings are designed to isolate two factors: whether the query is semantically expanded by an LLM, and whether the predicted attack type is provided as additional context. Experiment A represents the complete retrieval setting, where the LLM generates a behavior-level description using both the raw WAF log and the attack type predicted by the Contrastive Attack Type Analyzer. Experiment B removes the attack-type context from the LLM prompt. Experiment C directly uses the raw payload as the retrieval query without attack-type context. Experiment D prepends the predicted attack type to the raw payload, but does not perform LLM-based knowledge expansion.

The comparison between Experiments A and D evaluates the contribution of LLM-based knowledge expansion when attack-type context is available. Experiment D achieves a higher Top-1 score of 0.7780, indicating that directly appending the predicted attack type to the raw payload can retrieve a plausible first candidate. However, Experiment A achieves stronger Top-3, Top-5, and Top-10 accuracy, reaching 0.8635, 0.8710, and 0.8762, respectively. This suggests that LLM-based expansion does not simply optimize exact Top-1 matching; rather, it produces richer behavior-level descriptions that better align with the semantic distribution of CTI procedure descriptions and retrieve the correct technique within a compact candidate set.

The comparison between Experiments B and C further confirms the independent contribution of LLM-based expansion when attack-type context is absent. Using the raw payload alone performs poorly, with Top-3 and Top-5 accuracy both equal to 0.0778. After LLM-based expansion, Top-3 and Top-5 accuracy increase to 0.3102 and 0.5615, respectively. This shows that transforming syntax-heavy payloads into natural-language behavioral descriptions substantially reduces the representational mismatch between WAF logs and CTI records.

TABLE X
TTP RETRIEVAL PERFORMANCE USING DIFFERENT LLMs
FOR KNOWLEDGE EXPANSION

Model	Params.	Top-1	Top-3	Top-5	Top-10
Llama-3.1-8B-Instruct (default)	8B	0.5787	0.8635	0.8710	0.8762
Qwen3-8B	8B	0.3500	0.6143	0.6857	0.7614
microsoft/phi-4	14B	0.3386	0.6571	0.7014	0.7557
meta-llama/llama-3.1-70b-instruct	70B	0.2857	0.5443	0.6229	0.7786
z-ai/glm-4.5-air	(MoE)	0.2457	0.5757	0.6243	0.6886
google/gemma-3-12b-it	12B	0.2729	0.4271	0.4871	0.6214
openai/gpt-oss-20b	20B	0.2286	0.4586	0.5486	0.7029
deepseek/deepseek-chat-v3	(MoE)	0.2329	0.4743	0.5800	0.7129

The comparison between Experiments A and B evaluates the contribution of attack-type context within the LLM prompt. When the predicted attack type is included, Top-3 accuracy increases from 0.3102 to 0.8635, and Top-5 accuracy increases from 0.5615 to 0.8710. This demonstrates that attack-type information plays an important role in guiding the LLM to generate more precise and retrieval-oriented behavioral descriptions.

Finally, the comparison between Experiments D and C shows that attack-type context alone is also beneficial when no LLM expansion is used. Adding the predicted attack type to the raw payload increases Top-1 from 0.0777 to 0.7780 and Top-3 from 0.0778 to 0.7972. However, the complete setting in Experiment A provides the best Top-3, Top-5, and Top-10 performance for analyst-oriented candidate generation. Overall, these ablation results demonstrate that the proposed CTI-based retrieval module benefits from both components: LLM-based semantic expansion bridges the gap between raw payload syntax and CTI narratives, while the attack-type context predicted by the Contrastive Attack Type Analyzer guides the expansion toward the correct adversarial behavior.

1) *Comparison Across Different LLMs*: To further examine the effect of the LLM used for knowledge expansion, we compare eight LLMs, including Llama-3.1-8B-Instruct and Llama-3.1-70B-Instruct [58], Qwen3-8B [61], Phi-4 [62], GLM-4.5-Air [63], Gemma-3-12B-IT [64], GPT-OSS-20B [65], and DeepSeek-V3 [66], while keeping the remaining TTP retrieval pipeline unchanged. In each setting, the same attack-type prediction and CTI-based retrieval procedure are used, and only the LLM responsible for transforming the WAF evidence into a behavior-level description is replaced. Table X summarizes the resulting Top-*K* retrieval performance.

The comparison shows that increasing the parameter scale does not necessarily improve TTP retrieval performance. For example, the 70B Llama-3.1 model does not consistently outperform the smaller Qwen3-8B and Phi-4 models. Among the alternative models, Qwen3-8B achieves a Top-1 accuracy of 0.3500, while Phi-4 achieves a Top-3 accuracy of 0.6571. The default Llama-3.1-8B-Instruct model provides the strongest overall retrieval performance, reaching 0.8635, 0.8710, and 0.8762 for Top-3, Top-5, and Top-10 accuracy, respectively. These results indicate that parameter scale alone is not a reliable predictor of retrieval quality; the ability of the LLM to produce behavior-level descriptions that align with the CTI corpus is also important. The results also support the modular design of the knowledge expansion stage. Different LLMs can

be substituted without retraining the attack-type classifier or reconstructing the downstream retrieval pipeline, although the resulting retrieval quality varies across models.

2) *Case Study: Semantic Overlap of TTPs*: To further elucidate the validity of using Top-*K* metrics, we examine a representative case from our test set involving a complex SQL injection payload. This payload combined a UNION SELECT query targeting the SYS.USER and PASSWORD tables with extensive URL encoding and comment-based obfuscation. In the ground-truth annotation, the attack corresponds to T1190 (Exploit Public-Facing Application). However, the retrieval system assigned T1110 (Brute Force) as the Top-1 prediction, while T1190 appeared as the Top-2 result.

Although the Top-1 prediction does not strictly match the ground-truth T1190 label, it reflects a semantically adjacent retrieval result rather than a direct replacement for the reference label. The payload exploits a public-facing application through SQL injection, which corresponds to T1190, but it also explicitly targets user and password fields. This credential-related context can increase the similarity to CTI descriptions associated with credential-access behaviors, including T1110. This example illustrates that Top-3 and Top-5 accuracy, particularly when LLM-based expansion and attack-type context are used together, provides a more meaningful indicator of operational utility for threat hunting than rigid Top-1 correspondence.

3) *Comparison With Direct TTP Attribution Baselines*: To further examine whether the proposed pipeline provides advantages over direct TTP attribution baselines, we compare it with two representative baseline methods. The first baseline is an OWASP CRS Rule-ID Heuristic that maps attack-specific CRS rule IDs to predefined TTP candidates. Although OWASP CRS is not designed for ATT&CK mapping, its rule IDs, rule tags, and rule messages still contain explicit attack semantics. Because this mapping depends only on static CRS rule semantics, it can be constructed offline and reused whenever the corresponding rule is triggered. Therefore, this baseline represents a practical rule-driven approach for estimating how much TTP-level visibility can be obtained from OWASP CRS metadata alone.

The second baseline is a supervised embedding-based TTP classifier using SecureBERT2.0. In this baseline, WAF logs are encoded using SecureBERT2.0, and a fully connected classification head is trained to directly predict TTP labels. This baseline evaluates whether a cybersecurity-domain encoder can solve the TTP attribution task without the proposed decomposition into Contrastive Attack Type Analysis, Knowledge Expansion, and CTI-based TTP Retrieval.

As shown in Table XI, the OWASP CRS Rule-ID Heuristic is effective when a log triggers a specific mappable CRS rule. However, its applicability is limited because only 13.59% of evaluated logs contain attack-specific CRS rule IDs that can be directly mapped to TTP candidates. This confirms that OWASP CRS metadata provides useful TTP cues, but relying on CRS metadata alone leaves most logs outside rule-driven TTP attribution.

The supervised SecureBERT2.0 baseline covers the complete evaluated attribution space because it directly predicts a

TABLE XI
COMPARISON WITH DIRECT TTP ATTRIBUTION BASELINES

Method	Attribution Strategy	Coverage	Main Result
OWASP CRS Rule-ID Heuristic	Maps CRS rule IDs to predefined TTP candidates based on CRS rule semantics	13.59%	Correct on mappable logs
Supervised SecureBERT2.0	SecureBERT2.0 embedding + FC classifier for direct single-label TTP classification	100%	Macro F1 = 0.1940; Weighted F1 = 0.8405
Our Proposed Method	Contrastive Attack Type Analysis + LLM-based Knowledge Expansion + CTI-based TTP Retrieval	100%	Top-3 Accuracy = 0.8635

TTP label for each input log. However, its macro-F1 score is only 0.1940, despite a weighted-F1 score of 0.8405. This gap indicates that direct single-label TTP classification is highly affected by sparse and imbalanced labels. In contrast, the proposed method formulates TTP attribution as CTI-grounded retrieval and achieves 0.8635 Top-3 accuracy. These results demonstrate that the proposed decomposition is more suitable for WAF-log-based TTP attribution than either the OWASP CRS Rule-ID Heuristic or direct supervised TTP classification.

D. Operational Analysis: TTP Coverage vs. OWASP CRS (RQ4)

Finally, we evaluate the operational value of our framework by comparing its TTP coverage with a heuristic baseline derived from OWASP CRS rule semantics. Because OWASP CRS does not provide native ATT&CK labels, this baseline is used only to estimate the TTP-level visibility that can be inferred from CRS metadata.

1) *Mapping Methodology*: To ensure a fair comparison, we established the TTP coverage baselines for both approaches using a consistent annotation methodology supported by a Large Language Model (GPT-4o).

For our approach, we applied the annotation pipeline to the augmented dataset containing the diverse attack variants generated through MCTS. GPT-4o was used to analyze the semantic context of each WAF log and assign the most plausible ATT&CK Technique ID. To ensure high fidelity, all initial assignments were subsequently reviewed and validated by security experts to confirm the logical consistency of the attribution. This rigorous process yielded a verified set of 206 unique TTPs.

For the OWASP CRS, we parsed the complete rule definitions of version 3.3 and extracted the raw rule logic, including regular expressions and rule tags, which were then provided to GPT-4o. Because CRS rules do not include native TTP labels, the model interpreted the defensive intent of each rule and mapped it to an appropriate ATT&CK technique, producing both a confidence score and a rationale. After discarding mappings with insufficient confidence, the CRS was found to cover only 28 unique TTPs.

2) *Coverage Comparison and Operational Reliability*: Although our framework identifies 206 unique TTPs, not all techniques appear with equal frequency across the dataset. These techniques are obtained by aggregating the ATT&CK mappings produced across all evaluated WAF logs and removing duplicate technique IDs. As shown in Figure 8, the

TABLE XII
TTP COVERAGE COMPARISON: OWASP CRS VS. OUR APPROACH

Metric	OWASP CRS	Our Approach	Improvement
Total Unique TTPs Identified	28	206	7.4×
High-Confidence TTPs ($t \geq 10$)	28	75	2.68×

resulting distribution is highly skewed: many logs are repeatedly associated with common web-related techniques, while other techniques appear less frequently and form a long-tail pattern. To distinguish between robust, sustained behavioral patterns and potentially sporadic noise, we apply a minimum frequency threshold $t \geq 10$. A technique is considered operationally reliable if it appears in at least t distinct WAF logs. This conservative threshold ensures that retained techniques represent recurrent attack behaviors rather than one-off anomalies or misclassifications.

Table XII presents a multi-tiered comparison. The OWASP CRS Rule-ID Heuristic identifies 28 unique ATT&CK techniques inferred from CRS rule semantics, while our framework identifies 206 unique techniques from the verified TTP attribution space, corresponding to a 7.4× increase in TTP-level visibility. We focus our main benchmark on the high-confidence set ($t \geq 10$) to ensure that the comparison reflects recurrent and operationally reliable behaviors rather than one-off assignments.

Even after applying this strict frequency filter, our approach retains 75 robust techniques, significantly exceeding the 28 techniques inferred from the OWASP CRS Rule-ID Heuristic. This represents a 2.68× improvement in reliable operational visibility. The results underscore that the semantic learning engine detects a wider spectrum of validated tactical behaviors than regex-based signatures.

In addition, we analyze the intersection between the OWASP CRS Rule-ID Heuristic and our high-confidence set. Our filtered model ($t \geq 10$) successfully covers 60.7% (17 out of 28) of the techniques inferred from CRS rule semantics. The remaining 11 CRS-inferred techniques absent from our high-confidence set include:

- **Low-Frequency Payload Vectors**: Techniques such as T1059 (Command and Scripting Interpreter) and T1499 (Endpoint Denial of Service) were identified by our model but fell below the strict frequency threshold ($t < 10$), indicating they are present but rare in the test dataset.
- **Non-Payload Mechanisms**: Techniques like T1199 (Trusted Relationship), T1131 (Authentication Package),

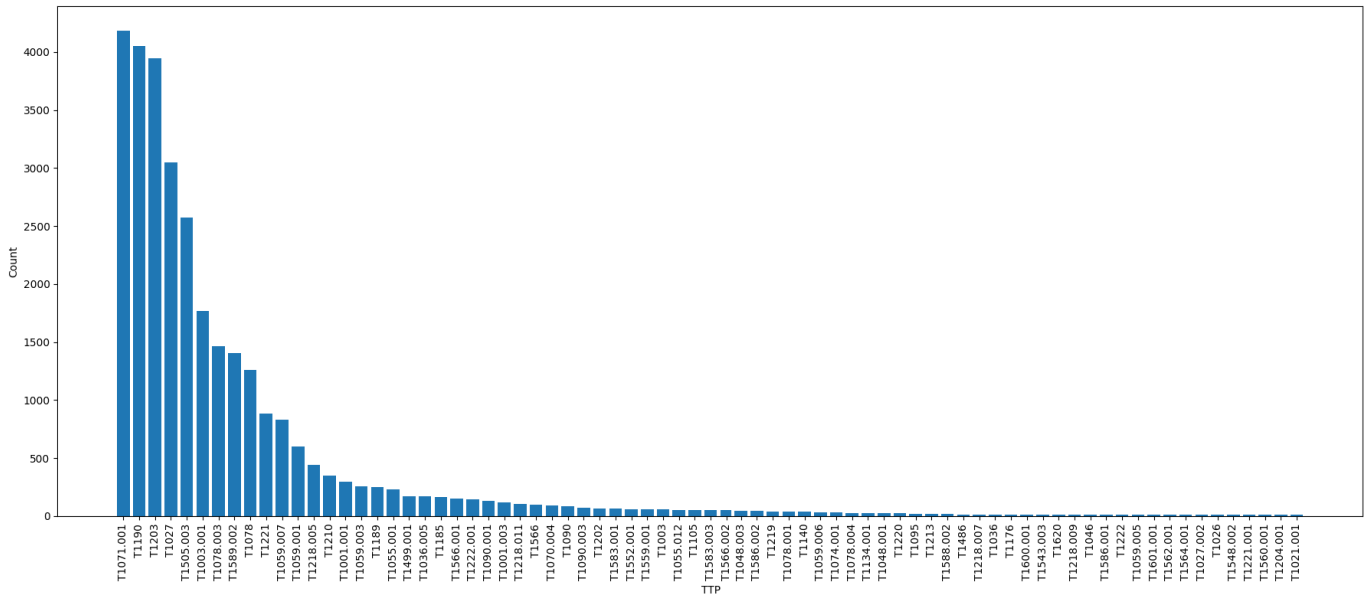


Fig. 8. Frequency distribution of high-confidence ATT&CK techniques.

and T1592 (Gather Victim Host Information) rely on network-level attributes (e.g., specific headers or handshake anomalies) rather than the payload injection patterns our MCTS engine is optimized to generate.

This gap analysis highlights a strategic trade-off: while strict thresholding slightly reduces the overlap with legacy rules (by filtering out rare matches), it provides a far more reliable signal for the 75 validated techniques that constitute the majority of the threat landscape.

3) Granularity of Rule-Level and Log-Level Attribution:

The preceding coverage analysis considers the TTP information obtainable from CRS rule semantics. However, requests triggering the same rule may contain different payload-level behaviors. To quantify this variation, we grouped a balanced subset of 700 external ModSecurity logs according to their triggered CRS rule signatures, where a rule signature is defined as the set of attack-specific CRS rule IDs associated with an individual event. We then examined the diversity of Top-1 techniques retrieved from logs sharing the same rule signature.

Among the 59 distinct rule signatures observed in the 700 logs, 29 (49.2%) are associated with more than one distinct Top-1 retrieved technique. These rule signatures account for 90.3% of the evaluated logs. Table XIII presents representative signatures with high technique multiplicity. For example, 120 logs triggering rule 930130 yield 10 distinct Top-1 techniques, while 65 logs sharing the rule signature 932130/932150 yield six distinct Top-1 techniques. These results show that requests matched by the same detection logic can contain substantially different ATT&CK-relevant behavioral information.

Representative events further illustrate this variation. For CRS rule 930130 (Restricted File Access), the fixed rule-derived candidates are T1005 and T1190. However, a request targeting `/.aws/credentials` retrieves T1552 among its Top-3 candidates, while `/.env.local` retrieves T1552, T1078, and T1110. In contrast, `/.git/config` retrieves T1105 and T1005. Although these requests trigger the same

TABLE XIII
REPRESENTATIVE CRS RULE SIGNATURES ASSOCIATED WITH MULTIPLE
LOG-LEVEL TOP-1 ATT&CK TECHNIQUES

CRS Rule Signature	Rule Semantics	Logs	Distinct Top-1 TTPs
930130	Restricted File Access (LFI)	120	10
941110	XSS Script Tag Vector	39	10
941120	XSS Event Handler Vector	52	9
932130, 932150	Unix Shell Expression / Command	65	6
930120, 932160	OS File Access + Unix Shell Code	10	6
942100	SQLi (libinjection)	38	5

CRS rule, their resource-specific content exposes different ATT&CK-relevant behaviors that cannot be distinguished from the rule ID alone.

A similar pattern is observed for command-injection events sharing rules 932130 and 932150, whose rule-derived candidates are T1059 and T1190. A `curl`-based request retrieves T1105 among its Top-3 candidates, a web-shell request retrieves T1505, and another remote-execution request retrieves T1210. These examples illustrate that CRS rule semantics provide reusable detection-level context, while the content of an individual request can expose additional behavioral information associated with specific resources, commands, and payload actions. Consequently, events sharing the same CRS rule signature may support different ATT&CK candidate sets when their event-specific content is considered.

E. External Generalization Evaluation (RQ5)

We further evaluate the trained framework on an independent real-world ModSecurity dataset described in

TABLE XIV
EXTERNAL ATTACK-TYPE CLASSIFICATION PERFORMANCE
ON REAL-WORLD MODSECURITY LOGS

Class	Precision	Recall	F1-score	Support
CI	1.0000	0.9074	0.9515	3,036
LFI	1.0000	0.9954	0.9977	49,108
MFU	0.9272	1.0000	0.9622	140
SQLi	0.9995	0.9367	0.9670	4,057
XSS	0.8879	0.7586	0.8182	261
Accuracy			0.9854	56,602
Macro Avg	0.9629	0.9196	0.9393	56,602
Weighted Avg	0.9993	0.9854	0.9921	56,602

TABLE XV
EXTERNAL TTP RETRIEVAL PERFORMANCE ON
REAL-WORLD MODSECURITY LOGS

Dataset	Top@1	Top@3	Top@5	Top@10
External ModSecurity logs	0.3529	0.6943	0.8200	0.9043

Section VI-B.5. The external logs were collected from a production web server and therefore differ from the controlled DVWA-based logs used in our primary evaluation in terms of traffic source, request diversity, and rule-trigger distribution. This setting allows us to examine whether the learned representations remain effective when applied to naturally occurring ModSecurity audit logs outside the data generation pipeline used for model training.

1) *External Attack-Type Classification*: Table XIV reports the attack-type classification performance on the external dataset. The evaluation includes five attack classes that can be reliably mapped from the external CRS rule IDs: CI, LFI, MFU, SQLi, and XSS. Because the external rule-trigger distribution is highly imbalanced, with LFI accounting for the majority of samples, we use macro-F1 as the primary summary metric. The classifier achieved a macro-F1 score of 0.9393 across 56,602 external logs, while the overall accuracy and weighted-F1 score reached 0.9854 and 0.9921, respectively. These results indicate that the CodeBERT-CL + K-means classifier maintains stable performance on independently collected ModSecurity logs.

The results reveal differences across attack types, which are reflected in the macro-F1 score of 0.9393. LFI achieves the highest F1-score of 0.9977 because the external CRS rules associated with restricted file access and path traversal produce relatively consistent structural patterns. SQLi and CI also remain robust, with F1-scores of 0.9670 and 0.9515, respectively, indicating that the learned representation transfers well to injection and command-execution patterns observed in external ModSecurity logs. XSS obtains a lower F1-score of 0.8182, which is likely due to its smaller support and the diversity of event-handler, attribute-based, JavaScript URI, and blacklist-triggered XSS patterns in the external CRS logs.

2) *External TTP Retrieval*: We further evaluated the CTI-based TTP retrieval module on a balanced subset of 700 externally labeled ModSecurity logs with multi-ground-truth

ATT&CK reference labels derived from Table VI. Specifically, we sampled 140 logs from each of the five externally mappable attack classes, including CI, LFI, MFU, SQLi, and XSS. This balanced setting avoids allowing high-frequency classes, such as LFI, to dominate the retrieval evaluation. Table XV reports the Top- K retrieval performance.

This external evaluation complements the controlled DVWA-based experiments by testing the retrieval module under naturally occurring ModSecurity rule-trigger distributions. The controlled setting provides precise validation against known vulnerable endpoints, whereas the external dataset reflects production logs with more diverse request patterns and rule-trigger behaviors. Although Top- K accuracy naturally increases as the candidate set becomes larger, the retrieval module retains the correct ATT&CK technique within a compact candidate set in most cases, achieving 0.6943 for Top-3, 0.8200 for Top-5, and 0.9043 for Top-10. This result indicates that the retrieval output remains useful for analyst triage even when exact Top-1 attribution is difficult under external multi-ground-truth labeling.

3) *Qualitative Inspection of Labeled Real-World Logs*: To complement the quantitative external evaluation, we further conducted a qualitative inspection of representative labeled real-world ModSecurity logs. This inspection provides a concrete view of how the proposed framework behaves under realistic operational conditions, where WAF logs are noisy, partially encoded, and often semantically ambiguous. The preceding RQ5 experiments report the formal external classification and retrieval results, and this subsection further illustrates representative examples and semantic interpretations.

For each inspected case, we examined the sanitized payload pattern, triggered CRS rule semantics, CRS-derived reference ATT&CK label, and retrieved ATT&CK candidates. Table XVI presents five representative cases selected from the externally mappable attack classes: SQLi, XSS, CI, LFI, and MFU. The reference ATT&CK labels follow the same CRS-derived external evaluation protocol used in Table VI. The qualitative interpretation further discusses cases where the retrieved candidates reflect alternative but operationally plausible behaviors.

The representative cases reveal three main qualitative findings. First, SQLi and CI show relatively direct semantic chains from CRS rule semantics to behavior-level interpretation and ATT&CK retrieval. For SQLi, sleep-based and union-style payloads are often abstracted as exploitation attempts against public-facing web applications, leading to retrieval results that include T1190. For CI, shell expressions and curl-based command execution are consistently associated with command execution behavior, and T1059 appears among the retrieved candidates. These cases demonstrate strong alignment between WAF-level evidence and ATT&CK-level behavioral interpretation.

Second, XSS and LFI reveal different forms of semantic dispersion. For XSS, the retrieved candidates include T1190, which is consistent with the CRS-derived reference label used in the external evaluation. However, real-world XSS payloads often involve browser-side JavaScript execution, event

TABLE XVI
REPRESENTATIVE CASES FROM QUALITATIVE INSPECTION OF LABELED REAL-WORLD MODSECURITY LOGS

Class	WAF Evidence	ATT&CK Mapping
SQLi	Payload: ?email=...&field=5559f5y21mD' OR 937=(SELECT 937 FROM PG_SLEEP(15))-- CRS Semantics: 942100 SQL Injection (libinjection); 942160 Blind SQLi (sleep)	Ref.: T1190 Retrieved: T1190, T1110
XSS	Payload: ?where=place"><svg+onload=confirm(document.domain)> CRS Semantics: 941120 XSS Event Handler Vector	Ref.: T1190 Retrieved: T1190, T1059, T1566
CI	Payload: GET /exec.php?exec=bash -c "\$(curl -fsSL https://[C2-URL])" CRS Semantics: 932130 Unix Shell Expression; 932150 Direct Unix Command	Ref.: T1059, T1190 Retrieved: T1090, T1059, T1105
LFI	Payload: GET /.git/config CRS Semantics: 930130 Restricted File Access	Ref.: T1005, T1190 Retrieved: T1105, T1005
MFU	Payload: POST /class-plugin-installer-skin-wp.php (multipart PHP upload) CRS Semantics: 933110 PHP Script File Upload	Ref.: T1505.003, T1190 Retrieved: T1071, T1041

handlers, and delivery-related patterns. As a result, the retrieval output may also include execution- or delivery-related techniques such as T1059 and T1566. This indicates that the framework can preserve the exploitation-level interpretation of XSS while also capturing additional behavioral semantics beyond the CRS-derived reference label. For LFI, the retrieved candidates include T1005, which is one of the CRS-derived reference labels. Nevertheless, LFI events can still be interpreted from multiple perspectives, including public-facing application exploitation, data access, restricted file access, or file and directory discovery. This highlights an inherent ambiguity when mapping web-layer WAF events to enterprise ATT&CK techniques.

Third, MFU illustrates a limitation of consequence modeling in the retrieval stage. Multipart PHP upload requests contain HTTP transfer artifacts and code-like tokens, but their security implication depends on the downstream consequence of deploying a server-side script or web shell. In some cases, the resulting retrieval representation may emphasize transport behavior rather than persistence or server-side execution capability, causing retrieved candidates to drift toward communication- or transfer-related techniques such as T1071 and T1041. This suggests that future work should incorporate consequence-aware prompting or class-conditional reranking to better preserve the post-exploitation implication of malicious file upload events.

Overall, this qualitative inspection complements the aggregate external metrics by explaining why semantic mismatches may occur in realistic ModSecurity logs. The observations suggest that SQLi and CI often provide clearer log-to-ATT&CK semantic chains, whereas XSS, LFI, and MFU require more careful interpretation at the TTP retrieval stage due to payload diversity, ATT&CK label ambiguity, and downstream consequence modeling.

F. Runtime and Resource Overhead Analysis (RQ6)

To address the operational feasibility of the proposed framework in high-volume SOC environments, we further profiled the runtime and resource overhead of the inference pipeline. The benchmark was conducted on the same server described

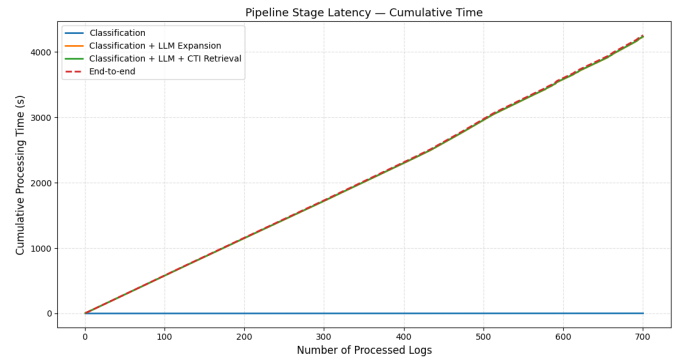


Fig. 9. Cumulative processing time by pipeline stage.

in Section VI-A, running Ubuntu 20.04×64 with an Intel Core i7-11700 CPU @ 2.50 GHz, 48 GB RAM, and an NVIDIA RTX A5000 GPU. The runtime benchmark focuses on inference rather than model training, and all measurements were collected using a single sequential worker without batching, caching, or parallel log processing.

As shown in Table XVII, the lightweight stages of the pipeline exhibit high throughput. Parsing only reaches 35.828 logs/s, corresponding to approximately 3.10 million logs per day, while the BERT-based defensive detection reaches 32.783 logs/s, corresponding to approximately 2.83 million logs per day. Raw retrieval without LLM-based expansion remains similarly efficient, processing 27.410 logs/s, or approximately 2.37 million logs per day. These results indicate that the non-LLM components can support million-scale daily WAF log processing under a single-worker setting.

In contrast, the full pipeline with local LLM-based knowledge expansion requires an average of 6.0755 seconds per log, corresponding to 0.165 logs/s or approximately 14,221 logs per day. Figure 9 further shows that cumulative processing time is dominated by the LLM expansion stage, while classification and retrieval introduce comparatively small overhead. These results indicate that local LLM expansion is the primary computational bottleneck in the end-to-end pipeline, especially

TABLE XVII
RUNTIME BENCHMARK SUMMARY

Setting	Mean Latency (s/log)	Median Latency (s/log)	P95 Latency (s/log)	Throughput (logs/s)	Est. Logs/day
Parsing only	0.0279	0.0279	0.0279	35.828	3,095,519
Classification	0.0305	0.0305	0.0305	32.783	2,832,482
Raw retrieval without LLM	0.0365	0.0363	0.0370	27.410	2,368,220
Full pipeline with LLM expansion	6.0755	5.7835	7.1434	0.165	14,221

TABLE XVIII
RESOURCE BENCHMARK SUMMARY

Item	Value
CTI records	18,088
Embedding dimension	384
Vector database disk size	58 MB
Runtime memory after loading	~255 MB
Peak CPU RAM	~5.99 GB
Peak GPU VRAM	~18.5 GB
Index backend	ChromaDB (HNSW via hnswlib)

when applied at per-log granularity in high-volume WAF environments.

Therefore, the proposed framework is not intended to invoke LLM-based expansion for every incoming WAF record. For alerts whose rule semantics already provide sufficient TTP-level context, precomputed rule-derived mappings can also provide a lightweight attribution option without additional semantic expansion. Instead, it should be deployed as a staged enrichment pipeline. The lightweight parsing, classification, and raw retrieval stages can first process high-volume WAF traffic and prioritize suspicious or ambiguous logs. LLM-based knowledge expansion can then be selectively applied to high-risk events, analyst-requested cases, or representative samples from repeated log clusters. In addition, caching repeated or near-duplicate WAF logs at the expanded-description or retrieval-result level can further reduce redundant LLM invocations. This deployment strategy preserves the operational feasibility of the framework while retaining TTP-level semantic enrichment for logs requiring deeper investigation.

Table XVIII summarizes the resource overhead. The CTI vector database contains 18,088 records with 384-dimensional embeddings and occupies only 58 MB on disk, requiring approximately 255 MB of runtime memory after loading. The peak GPU memory usage is approximately 18.5 GB, primarily due to local LLM inference rather than vector retrieval. These results indicate that the CTI retrieval index itself is lightweight, while GPU dependency is mainly introduced by the local LLM expansion module. We also observed that replacing the local LLM with an API-based LLM can reduce wall-clock latency; however, this configuration is not used as the primary benchmark because it depends on external service latency, rate limits, cost, and organizational privacy policies.

VIII. DISCUSSION

A. Key Findings and Implications

This study demonstrates that integrating semantic learning with traditional WAF deployments can substantially enhance

threat visibility and operational intelligence. Three principal findings emerge from our experimental evaluation.

First, the MCTS-based payload generation framework with semantic verification successfully addresses the data scarcity challenge that has long plagued WAF research. The resulting data comprise 175,509 payload samples and semantic-verification pairs, including 83,509 payload samples for attack classification and 92,000 original-mutated pairs for semantic verification. With a semantic verification accuracy of 99.98%, these results demonstrate that structure-aware mutation can generate diverse yet operationally valid attack variants while preserving the adversarial intent of their seed payloads, a critical requirement for training robust detection models.

Second, the CodeBERT-based contrastive learning framework achieves 99.94% multi-class classification accuracy and a macro F1-score of 0.9989 across eight distinct traffic categories. This represents a significant advancement over binary classification approaches (malicious vs. benign) that dominate prior work. The consistently high precision and recall across the different attack types demonstrate that pre-trained code models can effectively capture the syntactic and semantic structures inherent in web exploitation attempts. Moreover, the integration of K-means clustering, despite a slight drop to an F1-score of 0.9938, confirms that the learned embedding space exhibits strong geometric separation, enabling cluster-based identification of attack variants within the learned semantic regions. The external ModSecurity evaluation further supports this observation, with the classifier achieving a macro-F1 score of 93.93% on independently collected production logs despite the highly imbalanced external rule-trigger distribution.

Third, the CTI-grounded TTP retrieval mechanism successfully bridges the semantic gap between low-level WAF logs and high-level threat intelligence. By identifying 206 unique ATT&CK techniques, a $7.4\times$ increase over the 28 techniques inferred from the OWASP CRS Rule-ID Heuristic, our framework provides security analysts with substantially richer behavioral context. This result shows that OWASP CRS metadata contains useful attack semantics, but relying on CRS metadata alone is insufficient for comprehensive TTP visibility. Even after applying conservative frequency thresholding ($t \geq 10$) to ensure operational reliability, the system retains 75 robust techniques, representing a $2.68\times$ improvement in actionable threat visibility. The Top-5 retrieval recall of 87.10% further validates the effectiveness of LLM-based knowledge expansion in mapping technical artifacts to standardized adversarial behaviors. In the external retrieval setting, the correct ATT&CK technique is retained within the Top-5 candidates with an accuracy of 82.00%, indicating that

TABLE XIX
COMPREHENSIVE COMPARISON WITH STATE-OF-THE-ART

Dimension	Traditional Methods ^a	Embedding Methods ^b	TTP Extraction ^c	Our Approach
Detection Paradigm	Binary (Mal/Benign)	Binary (Mal/Benign)	Text→TTP	Multi-class + TTP
Semantic Understanding	Lexical (TF-IDF, n-grams)	Generic embeddings	NLP-based	CodeBERT + CL
Dataset Scale	Small (10K–100K)	Medium (100K–1M)	CTI reports	714K WAF logs
Ground Truth Validation	Assumed labels	Assumed labels	Manual annotation	Execution-validated
TTP Coverage	N/A	N/A	Limited (text-based)	206 techniques
Robustness to Evasion	Low (regex-based)	Medium	N/A	High (mutation-trained)
Operational Integration	Limited	Limited	Requires manual effort	Automated pipeline
Real-time Capability	Yes	Yes	No (batch processing)	Yes (modular design)

^a [29]–[31]^b [32]–[34]^c [27], [35]–[38], [40]

TABLE XX
REPRESENTATIVE SQLi MUTATION EXAMPLES

Mutation	Example
Case Swapping	or 1=1 → oR 1=1
Whitespace Substitution	or 1=1 → or/**/1=1 / or%0a1=1
Comment Injection	or 1=1 → /*foo*/or/*bar*/1=1
Comment Rewriting	/*foo*/or 1=1 → /*!1.png*/or 1=1
Integer Encoding	or 1=1 → or 0x1=0x1
Operator Swapping	or 1=1 → or 1 LIKE 1 / or 1 REGEXP 1
Logical Invariant	or 1=1 → or 1=1 AND 'a'='a'
Inline Comment	union select → /*!union*//*!50000select*/
Where Rewriting	where id=1 → where id=1 AND TRUE / where (select 0) or id=1
DML Substitution	or 1=1 → 1=1 / and name='foo' → && name='foo'
Tautology Substitution	'1'='1' → 2<>3 / 1=1 → rand()>=0

the retrieval pipeline remains useful under naturally occurring ModSecurity rule-trigger distributions.

From an operational perspective, these findings have direct implications for Security Operations Centers (SOCs). Traditional WAFs excel at blocking known attack patterns but provide limited insight into adversarial intent. Our framework transforms this paradigm by enabling behavior-centric threat hunting: instead of merely flagging “Inbound Anomaly Score Exceeded,” analysts receive precise TTP attributions such as T1190 (Exploit Public-Facing Application) or T1110 (Brute Force), complete with supporting evidence from retrieved CTI records. This contextual enrichment accelerates incident response, supports proactive threat modeling, and enables more effective alignment between defensive postures and evolving adversarial tactics.

B. Comparison With State-of-the-Art

Table XIX provides a comprehensive comparison between our approach and existing methods across multiple dimensions: detection paradigm, semantic understanding, dataset scale, TTP coverage, and operational applicability.

1) *Detection Paradigm and Granularity*: Unlike prior work that predominantly focuses on binary classification (malicious

vs. benign) [29], [30], [31], [32], our framework performs fine-grained multi-class categorization across eight traffic types. This distinction is critical for downstream TTP mapping: identifying an event as “malicious” provides minimal operational value, whereas categorizing it as SQLi, XSS, or Command Injection enables targeted investigation and response. Recent BERT-based approaches [33], [34] have improved malicious URL detection but remain confined to binary outcomes. In contrast, our use of CodeBERT with supervised contrastive learning explicitly learns discriminative representations for each attack class, enabling precise taxonomical attribution.

2) *Semantic Understanding and Robustness*: A key limitation of traditional feature engineering methods [29], [30], [31] is their reliance on static lexical patterns (n-grams, TF-IDF), which are brittle against obfuscation and evasion. Embedding-based methods [32], [33] improve generalization but typically lack the domain-specific tokenization required for code-like payloads. Our adoption of CodeBERT’s pre-trained tokenizer, combined with contrastive learning, directly addresses this gap. By training on MCTS-augmented data that includes extensive mutation variants (case swapping, comment injection, encoding obfuscation), our model learns invariant representations robust to adversarial perturbations. This is

TABLE XXI
REPRESENTATIVE MUTATION EXAMPLES FOR XSS, CI, LFI, AND MFU PAYLOADS

Class	Mutation	Example
XSS	Case Swapping	<code><script>alert(1)</script> → <Script>alert(1)</Script></code>
	Whitespace Substitution	<code> → <img%0asrc=x> / <img\tsrc=x></code>
	Comment Injection $\triangle$	<code><script>alert(1)</script> → <script> /*foo*/alert(1)</script></code>
	Tautology Substitution	<code><script>alert(1)</script> → <script>alert(1==1)</script></code>
CI	Case Swapping	<code>cat /etc/passwd → CAT /etc/passwd</code>
	Whitespace Substitution	<code>id;whoami → id%0awhoami / id\nwhoami</code>
	Comment Injection	<code>127.0.0.1;id → 127.0.0.1;#foo\nid</code>
	Operator Swapping $\triangle$	<code>&& → / ; → %0a</code>
	Logical Invariant $\triangle$	<code>127.0.0.1;id → 127.0.0.1 && true;id</code>
LFI	Case Swapping	<code>../etc/passwd → ../ETC/passwd</code>
	Whitespace Substitution	<code>?page=../etc/passwd → ?page=..%2fetc%2fpasswd</code>
	Integer Encoding	<code>../../../../etc/passwd (2-depth) → ../../../../../../etc/passwd (4-depth)</code>
	Tautology Substitution $\triangle$	<code>?page=../etc/passwd → ?page=../etc/passwd&a=1</code>
MFU	Case Swapping	<code>shell.php → shell.PHP / shell.Php</code>

reflected in the high classification accuracy (99.94%) and macro F1-score of 0.9989 despite the syntactic diversity of the test set.

3) *TTP Mapping and Intelligence Integration*: To the best of our knowledge, this is the first work to establish an automated pipeline from raw WAF logs to standardized MITRE ATT&CK techniques. Prior ATT&CK mapping research has considered unstructured CTI reports, structured detection rules such as Snort and Sigma, and operational security telemetry. Sigma rules can represent application-layer detections when relevant application or web events are available as their log sources. However, these approaches primarily operate on reusable detection-rule semantics or other security telemetry rather than performing event-level attribution directly from raw WAF audit records containing HTTP request content and WAF-specific metadata. Our CTI-grounded retrieval mechanism, which combines LLM-based knowledge expansion with embedding similarity search, uniquely bridges the representational gap between HTTP artifacts and high-level threat narratives. The 206 unique techniques identified demonstrate that semantic learning can uncover adversarial behaviors invisible to rule-based systems.

4) *Dataset Scale and Validity*: Many existing studies rely on small, synthetic, or outdated datasets (e.g., CSIC 2010 with 36,000 samples). Our framework operates on 714,162 WAF logs together with 175,509 payload samples and semantic-verification pairs across the attack-classification and semantic-verification datasets. Critically, the use of a controlled DVWA environment ensures that all training data consists of ground-truth verified exploitations rather than theoretical heuristics. This methodological rigor distinguishes our work from approaches that assume dataset labels without validation.

5) *Operational Applicability*: While several methods achieve high detection accuracy in controlled experiments, few address the practical requirements of SOC integration: explainability, real-time performance, and actionable outputs. Our framework’s modular design, separating payload generation, classification, and retrieval, enables incremental deployment.

The LLM-based expansion module operates in a zero-shot manner, allowing organizations to swap or upgrade models without retraining. Furthermore, the frequency-based thresholding mechanism (Section VII-D) provides a tunable trade-off between coverage breadth and detection reliability, accommodating diverse operational risk profiles.

C. Limitations

Despite the promising results, this study has several limitations that warrant careful consideration.

1. **Scope and Generalization Constraints**. The primary dataset used in this study was constructed in a controlled DVWA environment protected by ModSecurity with OWASP CRS v3.3, covering eight traffic categories: SQLi, XSS, CI, MFU, CSRF, LFI, BF, and Benign. This design provides reproducible ground-truth validation because each payload is executed against known vulnerable endpoints before WAF logs are collected. However, controlled environments cannot fully capture the heterogeneity of production deployments, including custom application logic, layered security controls, diverse technology stacks, evolving benign traffic, and deployment-specific CRS tuning. To partially address this limitation, we conducted an external evaluation using an independently collected thirty-day ModSecurity dataset from a production web server. The results provide additional evidence that the classifier and retrieval module remain effective under naturally occurring CRS rule-trigger distributions. Nevertheless, the external dataset remains ModSecurity/CRS-based and consists mainly of blocked or suspicious requests; it does not provide complete benign traffic ground truth or coverage of all attack types considered in our controlled testbed. The eight traffic classes should therefore be interpreted as the experimental classification scope rather than exhaustive coverage of web application attacks. The seven malicious classes were selected according to two constraints: the relevant attack evidence must be observable in the HTTP/WAF telemetry, and the attack must be reproducible against a known vulnerable endpoint to obtain reliable ground truth. This

distinction is particularly important for authorization-related vulnerabilities such as Broken Access Control and Broken Object Level Authorization (BOLA). Whether a request such as `GET /api/orders/12345` is malicious depends on application-side authorization state, including the authenticated user, resource ownership, and access-control policy. Such semantics are generally not encoded in the request syntax itself and therefore cannot be determined reliably from an isolated WAF record. Other omitted attack families are more naturally extensible within the present framework. Server-Side Request Forgery (SSRF), Server-Side Template Injection (SSTI), XML External Entity (XXE) injection, and deserialization or object-injection attacks can expose recognizable request-level artifacts. Supporting these classes would primarily require corresponding vulnerable endpoints, validated seed payloads, and attack-specific mutation rules, without changing the overall classification and attribution architecture. CSRF is already included in the current evaluation. The external evaluation further illustrates the practical difficulty of obtaining broad, reliable attack-type ground truth from operational WAF datasets. In the independently collected ModSecurity dataset, only five classes (CI, LFI, MFU, SQLi, and XSS) could be assigned sufficiently reliable reference labels from the available CRS rule information; CSRF and BF lacked suitable rule groups, and benign traffic ground truth was unavailable. Nevertheless, the classifier achieved a macro-F1 score of 0.9393 across 56,602 external logs, providing evidence that the learned classification capability generalizes beyond the controlled dataset. Future validation should therefore include additional payload-observable attack families, commercial WAF products, application-side context for authorization-dependent vulnerabilities, non-CRS telemetry, and production benign traffic.

2. Dependence on Large Language Models. The CTI-based TTP retrieval module relies on an LLM (Llama-3.1-8B-Instruct) for knowledge expansion. While this design offers flexibility, allowing the LLM to be swapped without retraining, it introduces three operational considerations. First, LLM-based expansion adds inference latency that may impact real-time analysis in high-throughput environments. Second, retrieval quality depends on the LLM selected for knowledge expansion. As shown in Section VII-C, different LLMs produce different Top- K retrieval performance even when the remaining retrieval pipeline is unchanged, indicating that the quality of the generated behavior-level descriptions affects their semantic alignment with the CTI corpus. Third, LLMs occasionally generate plausible but factually incorrect content (hallucination). Although our retrieval mechanism grounds results in a pre-validated CTI database (18,088 records), the expansion step could introduce speculative interpretations. We recommend implementing a confidence scoring mechanism that flags low-certainty expansions for manual review.

3. Performance Trade-offs. The framework exhibits two notable performance trade-offs. First, although the controlled evaluation achieves consistently high classification performance, the external ModSecurity evaluation exhibits greater class-dependent variation. In particular, XSS achieves a lower F1-score of 0.8182, which is likely due to its smaller support

and the diversity of event-handler, attribute-based, JavaScript URI, and blacklist-triggered XSS patterns in the external CRS logs. This indicates that strong performance in the controlled environment does not eliminate the challenges introduced by heterogeneous real-world traffic distributions. Second, applying a minimum frequency threshold ($t = 10$) for operational TTP coverage introduces a trade-off between reliability and completeness. While this filter ensures retained techniques represent sustained behavioral patterns, it excludes 131 low-frequency techniques (206 total minus 75 retained) that may include emerging or advanced persistent threats. Security teams should therefore consider maintaining separate analyses for high-confidence and exploratory technique sets.

4. Ground Truth Labeling Subjectivity. The TTP attribution process involves initial labeling using GPT-4o followed by expert validation. While this hybrid approach ensures high fidelity, it introduces potential subjectivity: different analysts may interpret ambiguous payloads differently, the LLM may exhibit inherent biases from its training data, and edge cases with multiple plausible TTP assignments may lack clear ground truth. Future work should quantify inter-annotator agreement using metrics such as Fleiss' kappa and explore automated consistency checks to strengthen ground-truth reliability.

IX. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper presents a novel framework that bridges the semantic gap between low-level WAF logs and high-level adversarial techniques, addressing a critical limitation in current web application defense. By integrating three advanced components—MCTS-based data augmentation, CodeBERT-driven semantic classification, and CTI-grounded TTP retrieval—we demonstrate that standard WAF deployments can be transformed into sources of actionable threat intelligence.

Experimental validation using 175,509 payload samples and semantic-verification pairs together with 714,162 WAF logs confirms the framework's effectiveness: 99.94% multi-class classification accuracy, 87.10% Top-5 TTP retrieval recall, and identification of 206 unique ATT&CK techniques (a $7.4\times$ increase over the 28 techniques inferred from the OWASP CRS Rule-ID Heuristic). These results establish that semantic learning, when properly grounded in execution-validated data and structured threat intelligence, enables Security Operations Centers to shift from reactive blocking to proactive, behavior-centric threat hunting. Complementary evaluation on an independent thirty-day production ModSecurity dataset further shows that the classifier maintains a macro-F1 score of 93.93% on external attack-type classification, while the retrieval module retains the correct ATT&CK technique within the Top-5 candidates with an accuracy of 82.00%.

The contributions of this work extend beyond empirical performance gains. Methodologically, we demonstrate that structure-aware mutation with semantic verification can generate large-scale, operationally valid adversarial datasets, addressing a long-standing challenge in WAF research. Architecturally, we show that decoupling contrastive representation

learning from clustering-based classification enables both high-precision categorization and unsupervised discovery of attack variants. Operationally, we establish the first automated pipeline for mapping application-layer logs to standardized TTPs, providing defenders with the contextual depth required for informed decision-making.

B. Future Work

Several promising directions emerge from this research:

1. Broader Cross-Environment Validation. Although this study includes external evaluation on production ModSecurity logs, broader validation across commercial WAF solutions (e.g., Imperva, F5, Cloudflare) and diverse application environments (CMS platforms, APIs, microservices) would further strengthen external validity. Additionally, incorporating emerging attack vectors such as SSRF, XXE, and deserialization attacks would enhance comprehensiveness. This expansion requires developing specialized mutation operators for MCTS and curating representative payloads, potentially through collaboration with bug bounty platforms or honeypot networks.

2. Adaptive Learning and CTI Currency. Web attack tactics evolve continuously, necessitating mechanisms to detect and adapt to distribution shifts. Future work should explore online learning strategies that incrementally update the classifier as new attack patterns emerge, possibly leveraging active learning to solicit analyst feedback on ambiguous cases. Additionally, automated CTI database refreshes synchronized with MITRE ATT&CK updates would ensure sustained alignment with the threat landscape.

3. Explainability for Operational Trust. While the framework provides TTP labels, security analysts would benefit from fine-grained explanations of why specific techniques were assigned. Integrating attention visualization mechanisms (e.g., highlighting influential tokens in WAF logs) or counterfactual explanations would enhance trust and facilitate investigative workflows.

By pursuing these avenues, the cybersecurity community can advance toward a future where web application firewalls not only block attacks but also provide rich, interpretable intelligence that empowers defenders to anticipate, understand, and respond to adversarial behaviors with unprecedented precision.

APPENDIX

REPRESENTATIVE MUTATION EXAMPLES FOR ATTACKSPECIFIC PAYLOADS

This appendix provides representative examples of the mutation operators used in the attack-specific payload generation process. These examples illustrate how the CFG-based mutation operators modify payload-level syntactic units for different attack classes. The symbol “△” follows Table IV and indicates context-dependent applicability. BF and CSRF are not listed because BF is generated using dictionary-based login attempts, while CSRF is represented through request-flow templates rather than CFG-based payload mutation.

REFERENCES

- [1] I. Darmawan, A. Nuridwan, A. Rahmatulloh, R. Gunawan, and R. Rizal, “Real-time web application firewall monitoring uses the OWASP CRS framework,” in *Proc. 9th Int. Conf. Informat. Comput. (ICIC)*, Oct. 2024, pp. 1–6.
- [2] M. Anuvashini, K. S. S. Bala, S. S. T. Sonti, and K. Jevitha, “An empirical study on the evaluation and enhancement of OWASP CRS (core rule set) in modsecurity,” *Comput. Secur.*, vol. 160, Jan. 2026, Art. no. 104714.
- [3] M. Sepczuk, “Dynamic web application firewall detection supported by cyber mimic defense approach,” *J. Netw. Comput. Appl.*, vol. 213, Apr. 2023, Art. no. 103596.
- [4] M. Vielberth, F. Böhm, I. Fichtinger, and G. Pernul, “Security operations center: A systematic study and open challenges,” *IEEE Access*, vol. 8, pp. 227756–227779, 2020.
- [5] L. Yu et al., “Detecting malicious web requests using an enhanced TextCNN,” in *Proc. IEEE 44th Annu. Comput., Softw., Appl. Conf. (COMPSAC)*, Jul. 2020, pp. 768–777.
- [6] J. R. Tadhani, V. Vekariya, V. Sorathiya, S. Alshathri, and W. El-Shafai, “Securing web applications against XSS and SQLi attacks using a novel deep learning approach,” *Sci. Rep.*, vol. 14, no. 1, p. 1803, Jan. 2024.
- [7] S. He, Y. Zhang, D. Liang, and P. Kumar Sharma, “An unsupervised malicious web request detection based on transformer and contrastive learning,” *IEEE Trans. Netw. Service Manage.*, vol. 22, no. 4, pp. 3281–3294, Aug. 2025.
- [8] V. Babaey and A. Ravindran, “GenSQLi: A generative artificial intelligence framework for automatically securing web application firewalls against structured query language injection attacks,” *Future Internet*, vol. 17, no. 1, p. 8, Dec. 2024.
- [9] Z. Iqbal Khan, M. Mazhar Afzal, and K. Naim Shamsi, “A comprehensive study on CIC-IDS2017 dataset for intrusion detection systems,” *Int. Res. J. Adv. Eng. Hub (IRJAEH)*, vol. 2, no. 2, pp. 254–260, Feb. 2024.
- [10] A. Kumar, K. Guleria, R. Chauhan, and D. Upadhyay, “Advancing intrusion detection with machine learning: Insights from the UNSW-NB15 dataset,” in *Proc. IEEE Int. Conf. Inf. Technol., Electron. Intell. Commun. Syst. (ICITEICS)*, Jun. 2024, pp. 1–5.
- [11] D. Chatziamanetoglou and K. Rantos, “Cyber threat intelligence on blockchain: A systematic literature review,” *Computers*, vol. 13, no. 3, p. 60, Feb. 2024.
- [12] SpiderLabs. (2023). *Modsecurity Web Application Firewall*. Accessed: Jan. 15, 2025. [Online]. Available: <https://github.com/SpiderLabs/ModSecurity>
- [13] OWASP Foundation. (2024). *OWASP Modsecurity Core Rule Set (CRS) V3.3.5*. Accessed: Jan. 15, 2025. [Online]. Available: <https://owasp.org>
- [14] DVWA Project. (2023). *Damn Vulnerable Web Application (DVWA)*. Accessed: Jan. 15, 2025. [Online]. Available: <https://github.com/digininja/DVWA>
- [15] Z. Qu, X. Ling, T. Wang, X. Chen, S. Ji, and C. Wu, “AdvSQLi: Generating adversarial SQL injections against real-world WAF-as-a-service,” *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 2623–2638, 2024.
- [16] K. Shen, P. Li, and X. Lin, “CLSESSP: Contrastive learning of sentence embedding with strong semantic prototypes,” *Knowl.-Based Syst.*, vol. 299, Sep. 2024, Art. no. 112053.
- [17] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, “K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data,” *Inf. Sci.*, vol. 622, pp. 178–210, Apr. 2023.
- [18] E. Aghaei, S. Jain, P. Arun, and A. Sambamoorthy, “SecureBERT 2.0: Advanced language model for cybersecurity intelligence,” 2025, *arXiv:2510.00240*.
- [19] G. Lucz and B. Forstner, “A thirty-day dataset of malicious HTTP requests blocked by OWASP ModSecurity on a production web server,” *Data*, vol. 10, no. 11, p. 186, Nov. 2025.
- [20] J. I. Christy Eunaicy and S. Suguna, “Web attack detection using deep learning models,” *Mater. Today: Proc.*, vol. 62, pp. 4806–4813, Jan. 2022.
- [21] S. Pillai and D. A. Sharma, “Hybrid unsupervised Web-attack detection and classification—A deep learning approach,” *Comput. Standards Interfaces*, vol. 86, Aug. 2023, Art. no. 103738.
- [22] S.-S. Chen, R.-H. Hwang, A. Ali, Y.-D. Lin, Y.-C. Wei, and T.-W. Pai, “Improving quality of indicators of compromise using STIX graphs,” *Comput. Secur.*, vol. 144, Sep. 2024, Art. no. 103972.

- [23] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (STIX)," *Mitre Corp.*, vol. 11, pp. 1–22, Jun. 2012.
- [24] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre att&ck: Design and philosophy," MITRE Corp., McLean, VA, USA, Tech. Rep., 2018.
- [25] F. O. Sönmez, C. Hankin, and P. Malacaria, "Attack dynamics: An automatic attack graph generation framework based on system topology, CAPEC, CWE, and CVE databases," *Comput. Secur.*, vol. 123, Dec. 2022, Art. no. 102938.
- [26] B. Al-Sada, A. Sadighian, and G. Oliveri, "MITRE ATT&CK: State of the art and way forward," *ACM Comput. Surveys*, vol. 57, no. 1, pp. 1–37, Jan. 2025.
- [27] L. Li, C. Huang, and J. Chen, "Automated discovery and mapping ATT&CK tactics and techniques for unstructured cyber threat intelligence," *Comput. Secur.*, vol. 140, May 2024, Art. no. 103815.
- [28] S.-S. Chen, T.-W. Pai, and C.-Y. Sun, "EnhanceCTI: Enhanced semantic filtering and feature extraction framework for industry-specific cyber threat intelligence," *Comput. Secur.*, vol. 158, Nov. 2025, Art. no. 104649. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404825003384>
- [29] E. Nowroozi, M. Mohammadi, and M. Conti, "An adversarial attack analysis on malicious advertisement URL detection framework," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 2, pp. 1332–1344, Jun. 2023.
- [30] J. Benita, C. Balaji, D. Kamalesh, D. S. Krishna, and C. M. Narasimharao, "Hybrid approach for malicious URL detection by integrating TF-IDF and random forest techniques," in *Proc. 4th Int. Conf. Mobile Netw. Wireless Commun. (ICMNWC)*, Dec. 2024, pp. 1–7.
- [31] G. Betarte, Á. Pardo, and R. Martínez, "Web application attacks detection using machine learning techniques," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 1065–1072.
- [32] M. Gniewkowski, H. Maciejewski, T. R. Surmacz, and W. Walentyńczak, "HTTP2vec: Embedding of HTTP requests for detection of anomalous traffic," 2021, *arXiv:2108.01763*.
- [33] Y. Li, Y. Wang, H. Xu, Z. Guo, Z. Cao, and L. Zhang, "URLBERT: A contrastive and adversarial pre-trained model for URL classification," 2024, *arXiv:2402.11495*.
- [34] A. El Mahdaoui, S. Lamsiyah, M. Janati Idrissi, H. Alami, Z. Yartaoui, and I. Berrada, "DomURLs_BERT: Pre-trained BERT-based model for malicious domains and URLs detection and classification," 2024, *arXiv:2409.09143*.
- [35] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu, "TTPDrill: Automatic and accurate extraction of threat actions from unstructured text of CTI sources," in *Proc. 33rd Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2017, pp. 103–115.
- [36] V. Legoy, M. Caselli, C. Seifert, and A. Peter, "Automated retrieval of ATT&CK tactics and techniques for cyber threat reports," 2020, *arXiv:2004.14322*.
- [37] N. Rani, B. Saha, V. Maurya, and S. Kumar Shukla, "TTPHunter: Automated extraction of actionable intelligence as TTPs from narrative threat reports," in *Proc. Australas. Comput. Sci. Week*, Jan. 2023, pp. 126–134.
- [38] M. T. Alam, D. Bhusal, Y. Park, and N. Rastogi, "Looking beyond IoCs: Automatically extracting attack patterns from external CTI," in *Proc. 26th Int. Symp. Res. Attacks, Intrusions Defenses*, Oct. 2023, pp. 92–108.
- [39] Y.-T. Huang et al., "MITREtrieval: Retrieving MITRE techniques from unstructured threat reports by fusion of deep learning and ontology," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 4, pp. 4871–4887, Aug. 2024.
- [40] Y.-C. Lai, C.-L. Yu, M.-L. Liao, Y.-S. Lin, Y.-C. Chang, and J.-L. Chen, "An intelligence defense system with SNORT rules," in *Proc. 25th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2023, pp. 249–254.
- [41] N. Daniel et al., "Labeling network intrusion detection system (NIDS) rules with MITRE ATT&CK techniques: Machine learning vs. large language models," *Big Data Cognit. Comput.*, vol. 9, no. 2, p. 23, Jan. 2025.
- [42] Y.-H. Lee, C. Han, M.-C. Peng, and T. Takahashi, "LLM-based multi-label mapping of snort rules to ATT&CK," in *Proc. IEEE Conf. Dependable Secure Comput. (DSC)*, Oct. 2025, pp. 1–6.
- [43] P. N. Wudali, M. Kravchik, E. Malul, P. A. Gandhi, Y. Elovici, and A. Shabtai, "Rule-ATT&CK mapper (RAM): Mapping SIEM rules to TTPs using LLMs," 2025, *arXiv:2502.02337*.
- [44] K. Kurniawan, A. Ekelhart, and E. Kiesling, "An ATT&CK-KG for linking cybersecurity attacks to adversary tactics and techniques," in *Proc. ISWC Posters, Demos Industry Tracks, Novel Ideas Ind. Practice (CEUR Workshop Proceedings)*, vol. 2980, 2021. [Online]. Available: <https://ceurws.org/Vol-2980/paper363.pdf>
- [45] S. Hans, S. Marsella, S. Hirschmann, and N. Gurney, "Security logs to ATT&CK insights: Leveraging LLMs for high-level threat understanding and cognitive trait inference," 2025, *arXiv:2510.20930*.
- [46] D. Appelt, C. D. Nguyen, L. C. Briand, and N. Alshahwan, "Automated testing for SQL injection vulnerabilities: An input mutation approach," in *Proc. Int. Symp. Softw. Test. Anal.*, Jul. 2014, pp. 259–269.
- [47] Y. Huang et al., "A mutation approach of detecting SQL injection vulnerabilities," in *Proc. Int. Conf. Cloud Comput. Secur. Cham*, Switzerland: Springer, 2017, pp. 175–188.
- [48] Y. Yuan, Y. Lu, K. Zhu, H. Huang, Y. Chen, and Y. Zhang, "SqlFuzz: Directed fuzzing for SQL injection vulnerability," *Electronics*, vol. 13, no. 15, p. 2946, Jul. 2024.
- [49] M. Heiderich, J. Schwenk, T. Frosch, J. Magazinius, and E. Z. Yang, "MXSS attacks: Attacking well-secured web-applications by using innerHTML mutations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 777–788.
- [50] E. Trickle et al., "Toss a fault to your witcher: Applying grey-box coverage-guided mutational fuzzing to detect SQL and command injection vulnerabilities," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 2658–2675.
- [51] S. Neef, L. Kleissner, and J.-P. Seifert, "What all the PHUZZ is about: A coverage-guided fuzzer for finding vulnerabilities in PHP Web applications," in *Proc. 19th ACM Asia Conf. Comput. Commun. Secur.*, Jul. 2024, pp. 1523–1538.
- [52] T. Lee, S. Wi, S. Lee, and S. Son, "FUSE: Finding file upload bugs via penetration testing," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–17, doi: [10.14722/ndss.2020.23126](https://doi.org/10.14722/ndss.2020.23126).
- [53] A. A. Hamza and R. J. Surayh Al-Janabi, "Detecting brute force attacks using machine learning," in *Proc. BIO Web Conf.*, vol. 97, 2024, p. 00045.
- [54] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," 2019, *arXiv:1910.01108*.
- [55] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2006, pp. 1735–1742.
- [56] Z. Feng et al., "CodeBERT: A pre-trained model for programming and natural languages," 2020, *arXiv:2002.08155*.
- [57] P. Khosla et al., "Supervised contrastive learning," in *Proc. NIPS*, 2020, pp. 18661–18673.
- [58] A. Grattafiori et al., "The llama 3 herd of models," 2024, *arXiv:2407.21783*.
- [59] Y. Fang, Y. Li, L. Liu, and C. Huang, "DeepXSS: Cross site scripting detection based on deep learning," in *Proc. Int. Conf. Comput. Artif. Intell.*, Mar. 2018, pp. 47–51.
- [60] Morzeux. (2019). *HttpParamsdataset. GitHub Repository*. Accessed: Dec. 2, 2024. [Online]. Available: <https://github.com/Morzeux/HttpParamsDataset>
- [61] A. Yang et al., "Qwen3 technical report," 2025, *arXiv:2505.09388*.
- [62] M. Abidin et al., "Phi-4 technical report," 2025, *arXiv:2412.08905*.
- [63] A. Zeng et al., "GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models," 2025, *arXiv:2508.06471*.
- [64] A. Kamath et al., "Gemma 3 technical report," 2025, *arXiv:2503.19786*.
- [65] S. Agarwal et al., "Gpt-oss-120b & gpt-oss-20b model card," 2025, *arXiv:2508.10925*.
- [66] A. Liu et al., "DeepSeek-V3 technical report," 2024, *arXiv:2412.19437*.



Sheng-Shan Chen (Graduate Student Member, IEEE) received the Ph.D. degree in computer science and information engineering from the National Taipei University of Technology (NTUT), Taipei, Taiwan. He has published his work in several venues, including *Computers and Security* journal and the proceedings of the IEEE Global Communications Conference (GLOBECOM), the International Conference on Medical and Health Informatics (ICMHI), and the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE). His research focuses on applying large language models to cyber threat intelligence analysis and healthcare security.



Ren-Hung Hwang (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Massachusetts, Amherst, MA, USA, in 1993. He is currently the Dean of the College of Artificial Intelligence, National Yang Ming Chiao Tung University (NYCU), Taiwan. Before joining NYCU, he was with National Chung Cheng University, Taiwan, from 1993 to 2022. His research interests include deep learning, wireless communications, networks security, the AIoT, and cloud/edge/fog computing. He has received multiple

Best Paper Awards from international conferences, including IEEE Ubi-Media, IEEE SC2, IEEE IUCC, and IEEE DataCom. He serves on the editorial boards of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS and *IEICE Transactions on Communications*.

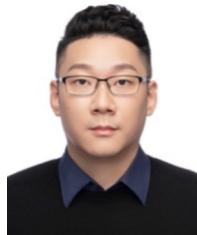


Tun-Wen Pai (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Duke University, Durham, NC, USA. He was the Department Chair from 2019 to 2022 and the Dean of the Office of International Affairs from 2022 to 2023 with the National Taipei University of Technology (Taipei Tech), Taipei, Taiwan. He is currently a Full Professor with Taipei Tech. His research interests include bioinformatics, medical informatics, machine learning, and data engineering. He is a member of ACM and the IEEE SMC Society.



Ying-Dar Lin (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California, Los Angeles (UCLA), in 1993. He was a Visiting Scholar with Cisco Systems from 2007 to 2008; the CEO of the Telecom Technology Center, Taiwan, from 2010 to 2011; and the Vice President of National Applied Research Laboratories (NARLabs), Taiwan, from 2017 to 2018. He is currently a Chair Professor of computer science with National Yang Ming Chiao Tung University (NYCU), Taiwan. His research interests include

networks security, wireless communications, networks softwarization, and machine learning for communications. He served as the Editor-in-Chief for IEEE COMMUNICATIONS SURVEYS AND TUTORIALS from 2017 to 2020.



Chin-Yu Sun (Member, IEEE) received the Ph.D. degree in computer science from National Tsing Hua University, Hsinchu, Taiwan, in 2021. He is currently an Assistant Professor with the Department of Computer Science and Information Engineering, National Taipei University of Technology, Taiwan, where he is also the Chief Officer of the Network Operation Section with the Computer and Network Center. His research interests include cybersecurity, hardware security, and quantum cryptography.