International Journal on Advanced Computer Theory and Engineering
ISSN: 2319 - 2526
Volume 15 Issue 2s (2026) | Pages: 105-109

Cyber Watch: A Zero-LLM Agentic Pipeline and Comprehensive Platform for Real-Time Cybersecurity Anomaly Detection

Divesh Bari[1], Ratnadeep Gurav[2], Shreyash Kolekar[3], Pranjali Kharate[4]
[1234]Department of Artificial Intelligence and Data Science Engineering GSMCOE
Email: diveshbari13@gmail.com, deepgurav2329@gmail.com, shreyashkolekar21@gmail.com, pranjalinkharate@gmail.com

Peer Review Information

***Type**: Article*
***Received**: 15 February 2026*
***Revised**: 18 March 2026*
***Accepted**: 11 April 2026*
***Published**: 20 May 2026*

Abstract

The escalating sophistication of modern cyber threats such as zero-day exploits and polymorphic malware necessitates advanced real-time detection and response mechanisms. While Large Language Models provide powerful reasoning capabilities for threat intelligence and contextual analysis, they introduce major limitations including high inference latency, expensive API dependency, and lack of deterministic explainability, making them unsuitable for split-second automated incident response systems. This paper introduces Cyber Watch, a production-ready Security Operations Center platform powered by a novel "Zero-LLM" agentic architecture. The proposed system replaces generative AI with Lang Graph-orchestrated deterministic NLP rules, Random Forest classifiers, PyTorch Neural Networks, and Explainable Decision Trees to achieve enterprise-grade autonomous threat analysis and mitigation. The architecture achieves sub-200 millisecond threat mitigation, real-time telemetry broadcasting, automated firewall response, and explainable threat analysis without relying on external generative AI APIs. Cyber Watch integrates Fast API backend middleware, a React frontend dashboard, and WebSocket telemetry infrastructure to provide scalable and responsive cybersecurity operations. Experimental results demonstrate high detection accuracy on the NSL-KDD dataset, secure log processing, automated mitigation execution, and UI broadcast latency below 50 milliseconds. The research proves that enterprise-grade autonomous security systems can achieve advanced AI-agent routing and real-time defense without the operational drawbacks of generative language models.

Keywords: Cybersecurity, Agentic Workflows, Explainable AI, Zero-LLM, Intrusion Detection, Role-Based Access Control.

How to Cite This Article

Bari, D., Gurav, R., Kolekar, S., Kharate, P. (2026). Cyber Watch: A Zero-LLM Agentic Pipeline and Comprehensive Platform for Real-Time Cybersecurity Anomaly Detection. *International Journal on Advanced Computer Theory and Engineering*, 15(2s), 105–109.

2319-2526/ © 2026 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0).

Introduction

The digital landscape is increasingly threatened by sophisticated cyberattacks capable of bypassing conventional rule-based security systems. Recent high-profile incidents demonstrate the urgent need for adaptive security, autonomous response systems, and AI-powered cyber defense mechanisms. As attackers increasingly leverage offensive AI, automated vulnerability discovery, and adaptive malware, defensive strategies must evolve to operate at machine speed and provide real-time mitigation capabilities. Recent developments in cybersecurity have led to the adoption of autonomous AI agents powered by Large Language Models and Retrieval-Augmented Generation frameworks to function as virtual Tier-1 and Tier-2 analysts. These systems significantly improved threat intelligence gathering, contextual analysis, and security automation capabilities. However, despite these advancements, LLM-based cybersecurity systems introduce several critical operational limitations when deployed in real-time incident response environments.

One of the most significant limitations is inference latency. Typical LLM inference operations require between three and eight seconds to return responses. Such delays are unacceptable during Distributed Denial-of-Service attacks, active intrusions, and rapid malware propagation scenarios where mitigation actions must occur within milliseconds. Additionally, generative models operate largely as black-box systems, making explainability and auditability extremely difficult. Security teams must be able to explain why a host was isolated, why an IP address was blocked, and why a mitigation action occurred in order to satisfy compliance frameworks and enterprise governance requirements. To overcome these limitations, this research presents CyberWatch, a fully containerized, enterprise-grade, full-stack cybersecurity platform built around a highly optimized Zero-LLM agentic pipeline. The platform wraps deterministic AI orchestration inside a modern React-based Security Operations Center application capable of delivering machine-speed autonomous threat mitigation without generative AI dependency. The primary objectives of this study include designing and evaluating a deterministic multi-agent AI pipeline using LangGraph routing without any LLM dependency, architecting a scalable FastAPI and React full-stack application with strict Role-Based Access Control, and validating platform efficiency under simulated enterprise workloads.

Literature review

The integration of Machine Learning and Deep Learning into cybersecurity has rapidly evolved from theoretical research into applied enterprise-grade architectures. Cybersecurity environments generate highly heterogeneous data involving users, hosts, files, and traffic flows. Recent studies increasingly utilize Heterogeneous Graph Neural Networks to capture semantic relationships and temporal dependencies in cybersecurity traffic and event streams.

Hybrid architectures combining Graph Neural Networks, Recurrent Neural Networks, and multi-head attention mechanisms have demonstrated strong capability in capturing both spatial dependencies and temporal attack patterns for intrusion detection tasks. Hierarchical cybersecurity systems integrating multi-modal data, scalar metrics, and raw text logs have also been proposed to improve contextual understanding and attack correlation. To balance detection accuracy and computational efficiency, researchers introduced multi-model voting systems and adversarially robust architectures capable of improving resilience against evolving cyber threats.

Retrieval-Augmented Generation frameworks have successfully connected Large Language Models with external databases for threat intelligence, dynamic reasoning, and knowledge retrieval. However, LLM integration introduces major risks including prompt injection attacks, high latency, and non-deterministic behavior. Several researchers emphasize that generative API latency and external dependency delays make LLMs unsuitable for immediate threat containment and millisecond-scale mitigation scenarios. Transparency and explainability remain additional concerns. Modern interpretability frameworks frequently rely on techniques such as TPOT and SHAP to improve understanding of model decisions.

The reviewed literature reveals a clear research gap. Although agentic workflows are highly effective for intelligent routing and threat triaging, the generative engines powering these systems remain too slow and too opaque for real-time autonomous mitigation. CyberWatch directly addresses this limitation by eliminating generative AI dependency and introducing a deterministic Zero-LLM architecture optimized for explainability, auditability, and low-latency threat response.

Methodology

The proposed research follows a comprehensive systems engineering methodology detailing AI engine construction, backend middleware architecture, real-time telemetry infrastructure, frontend presentation layer, and deployment-oriented orchestration. The overall framework is designed to support deterministic AI workflows, explainable threat analysis, and machine-speed automated mitigation.

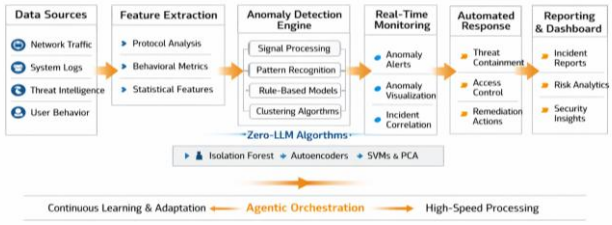

***Fig 1**: Cyber Watch Methodology – A Zero-LLM Agentic Pipeline for Real-Time Cybersecurity Anomaly Detection*

The zero-llm agentic pipeline

The core intelligence of cyberwatch functions entirely offline using a langgraph stategraph architecture to route a structured securitystate dictionary through four analytical phases. The architecture completely eliminates external llm api calls and removes all dependency on generative ai systems.

Agent 1: analysis

The analysis agent replaces llm-based context parsing using deterministic regular expressions and spacy named entity recognition. The agent extracts ipv4 addresses, privileged ports, failure counts, and indicators of compromise from security logs and telemetry streams. The extracted information is standardized into a 20-dimensional feature vector suitable for downstream machine learning processing. This stage enables instant parsing, deterministic extraction, and extremely low-latency preprocessing.

Agent 2: detection

The detection agent utilizes a dual-model ensemble consisting of a 300-estimator random forest classifier and a 4-layer pytorch neural network trained on the nsl-kdd dataset. Final classification is generated using weighted ensemble voting where the random forest contributes 40% and the neural network contributes 60% of the final prediction score. This ensemble approach improves detection accuracy, classification robustness, and generalization capability across diverse attack categories.

The ensemble weighting function can be represented as:

$$Prediction = 0.4(RF) + 0.6(NN)$$

Agent 3: prevention

To eliminate black-box decision making, the prevention agent uses a scikit-learn decision tree classifier with maximum depth equal to eight. The model maps detected threats against hard-coded nist sp 800-61 response matrices to generate exact mitigation strategies such as block_ip, isolate_host, and rate_limit. These actions are executed using os-native commands. The deterministic structure guarantees full explainability, auditability, and compliance compatibility.

Agent 4: reporting

The reporting agent replaces generative text generation using jinja2 templating and the reportlab library. Variables from the securitystate dictionary are injected into static templates to generate pdf reports, json payloads, and compliance documentation. This approach completely removes the possibility of ai hallucination and unverifiable report generation.

Backend middleware and api integration

The ai orchestration pipeline is exposed through a fastapi backend consisting of 36 rest endpoints. Since the prevention agent executes os-level mitigation commands, execution-path security becomes critically important. The api layer implements role-based access control and json web token authentication. Permissions are organized across four tiers: admin, analyst, collector, and viewer. This architecture ensures secure automated ingestion, restricted manual overrides, and cryptographically protected threat execution. Authorized users can manually invoke endpoints such as:

Post /api/v1/agents/analyze-threat

Real-time telemetry via websocket architecture

To complement the sub-millisecond ai pipeline, cyberwatch implements a bidirectional websocket manager. When anomalies are detected, the backend broadcasts structured payloads such as threat_detected and analysis_progress directly to the frontend interface.

The websocket infrastructure includes exponential backoff reconnection strategies and persistent heartbeat mechanisms with a heartbeat interval of 30 seconds. These mechanisms guarantee persistent connections, reliable telemetry delivery, and real-time synchronization across distributed clients.

Frontend dashboard and ui architecture

The frontend is implemented as a modern react application built using vite, typescript, and tailwind css. The application contains fifteen distinct ui components managed through zustand state stores. The dashboard component displays system metrics, telemetry summaries, and threat statistics. The alerts management panel provides color-coded severity badges, incident prioritization, and alert visualization. The logs viewer supports advanced pagination, filtering, timestamp visualization, and real-time updates. The overall interface maintains responsive design, cross-platform compatibility, and smooth rendering performance under heavy telemetry loads.

Containerization and quality assurance

The entire CyberWatch platform is orchestrated using Docker Compose. Multi-stage build optimization reduces the backend container size to approximately 500 MB and the frontend container size to approximately 150 MB. Traffic is securely routed through an Nginx reverse proxy to improve security and scalability.

The CI/CD pipeline executes Pytest tests, Vitest tests, integration testing, and unit testing on every codebase push. A minimum code coverage threshold of 80% is enforced to ensure platform reliability, stability, and production readiness.

Results And Performance Evaluation

The CyberWatch platform was evaluated across two primary dimensions: computational efficiency of the AI pipeline and operational responsiveness of the web platform.

Ai pipeline performance and latency

The detection agent models were trained using the nsl-kdd dataset. The ensemble approach maximized f1-score and threat classification accuracy while maintaining extremely low computational overhead.

Table I. Zero-llm ai pipeline execution latency

Pipeline stage	Processing mechanism	Average latency
Agent 1: analysis	Regex + spacy ner	< 5 ms
Agent 2: detection	Random forest + pytorch nn	120 ms
Agent 3: prevention	Decision tree classifier	< 2 ms
Agent 4: reporting	Jinja2 templating	45 ms
Total orchestration	Langgraph state execution	~172 ms

The complete end-to-end analytical pipeline executes within less than 200 milliseconds, allowing threats to be identified, classified, and mitigated within fractions of a second.

Platform operational metrics and ui responsiveness

During simulated enterprise workloads, the cyberwatch frontend successfully processed high telemetry volumes, continuous websocket updates, and real-time event rendering. The websocket broadcasting mechanismmaintained message latency between 10 and 50 milliseconds, ensuring immediate visual feedback and near-instant synchronization.

Table II. Dashboard visualization and telemetry metrics

Metric category	Observed capability	Ui representation
System status	99.9% uptime, active connections	Green status indicator
Active alerts	Dynamic tracking (e.g., 42 active)	Numerical badges
Analyzed logs	High volume (e.g., 15,234 ingested)	Real-time counters
Threat analysis	Deep inspection (e.g., brute force at 94%)	Detailed modal views

The react interface maintained 60 fps during intense log ingestion while rendering timestamped events, source ip addresses, and severity warnings without blocking the browser thread.

Discussion

The findings indicate that the cyberwatch architecture successfully resolves the operational bottlenecks associated with modern llm-based cybersecurity tools. By decoupling generative ai from the critical response path, the system achieves execution times below 200 milliseconds, representing a major improvement over traditional generative llm apis that frequently require three to eight seconds for inference. Because the analysis agent relies on deterministic regex patterns and rule-based parsing, the system becomes fundamentally immune to prompt injection attacks and llm manipulation exploits, which remain major vulnerabilities in generative ai security systems.

The use of scikit-learn decision trees within the prevention agent guarantees full explainability, deterministic reasoning, and auditable mitigation logic. Security teams can explicitly map mitigation actions to compliance frameworks such as soc2 and iso27001 policies. The exact mathematical branching logic can be exported to verify why an ip address was blocked, why a host was isolated, or why rate limiting was triggered. Such transparency is generally not achievable with black-box deep learning systems or generative llm architectures.

The integration of a react frontend, fastapi middleware, and websocket telemetry transforms cyberwatch from a theoretical ai model into a deployable, commercial-grade, enterprise-ready cybersecurity platform. The architecture abstracts away the complexity of pytorch tensors, langgraph routing, and machine learning orchestration while presenting analysts with color-coded alerts, real-time notifications, and actionable threat intelligence through an intuitive soc dashboard. Despite its advantages, the rigid zero-llm architecture introduces several trade-offs. Unlike llm and retrieval-augmented generation systems capable of dynamically understanding novel threat intelligence and unstructured reports, the deterministic analysis agent depends on predefined regex patterns and static parsing rules. Consequently, when entirely new log formats emerge, manual rule updates become necessary. Additionally, the random forest and pytorch neural network ensemble requires periodic retraining to maintain detection accuracy against evolving attack strategies, novel malware variants, and zero-day exploits. This introduces computational overhead and dataset maintenance requirements over time.

Conclusion

The integration of artificial intelligence into cybersecurity represents a necessary evolution for combating adaptive malware, automated attacks, and offensive ai systems. Traditional signature-based defenses are no longer sufficient against polymorphic threats, autonomous attack systems, and ai-driven exploit generation. Although large language models significantly improved threat intelligence and contextual reasoning, this research demonstrates their severe operational limitations for real-time incident response including high inference latency, prompt injection vulnerability, and lack of explainability. These weaknesses make generative ai unsuitable for split-second mitigation and deterministic autonomous defense systems. This paper introduced and validated cyberwatch, a comprehensive security operations center platform powered by a deterministic zero-llm architecture. The system demonstrates that sophisticated agent routing, ai orchestration, and threat automation can be achieved without generative ai or external llm apis. The architecture consists of four deterministic agents: regex and spacy ner for semantic extraction, a random forest and pytorch neural network ensemble for anomaly detection, an explainable decision tree prevention engine, and jinja2 templating for compliance-safe reporting. This architecture enables enterprise-grade defense, machine-speed mitigation, and human-auditable reasoning. The complete ai orchestration executes within less than 200 milliseconds, significantly outperforming current llm-based systems. The platform further integrates docker containerization, fastapi middleware, websocket telemetry, and a react frontend to provide scalability, real-time situational awareness, and analyst override capability. The websocket infrastructure delivers telemetry updates within less than 50 milliseconds, ensuring near-instant ui synchronization.

Future research directions

Future iterations of cyberwatch should integrate heterogeneous graph neural networks for multi-modal telemetry fusion, spatial relationship analysis, and advanced threat correlation. The detection agent could simultaneously analyze network traffic, endpoint cpu heuristics, and user behavior graphs to improve advanced persistent threat detection and coordinated attack analysis. Future architectures may additionally incorporate recurrent neural networks and long short-term memory layers to improve detection of slow-moving attacks, lateral movement, and temporal intrusion patterns. Capturing multi-day event sequences could significantly reduce false negatives and delayed detection. Another important future direction involves privacy-preserving federated learning. Instead of centralized retraining, decentralized security operations centers could collaboratively train the shared pytorch neural network without exchanging raw traffic data. Only model gradients would be exchanged, thereby improving global threat immunity while preserving organizational privacy and reducing centralized data exposure.

References

1. l. Jiang, r. Ryan, q. Li, and n. Ferdosian, "a survey of heterogeneous graph neural networks for cybersecurity anomaly detection," arxiv preprint arxiv:2510.26307, 2025.
2. j. Biradar, s. Shah, and t. Naik, "attention augmented gnn rnn-attention models for advanced cybersecurity intrusion detection," arxiv preprint arxiv:2510.25802, 2025.
3. Borah, m. T. Alam, and n. Rastogi, "adapting large language models to emerging cybersecurity using retrieval augmented generation," arxiv preprint arxiv:2510.27080, 2025.
4. r. Somani and a. K. Cherukuri, "large language models for cyber security," arxiv preprint arxiv:2511.04508, 2025.
5. p. B. Yakubu et al., "automated and explainable denial of service analysis for ai-driven intrusion detection systems," arxiv preprint arxiv:2511.04114, 2025.
6. c.-w. Chang et al., "scam shield: multi-model voting and fine-tuned llms against adversarial attacks," arxiv preprint arxiv:2511.01746, 2025.
7. t. Osinaike, a. Yetunde, and c. V. Onyenagubo, "a survey of ai-powered proactive threat-hunting techniques: challenges and future directions," *international journal for multidisciplinary research (ijfmr)*, vol. 6, no. 6, pp. 21–22, 2024.
8. j. Zhang, z. Zhu, j. Deng, y. Li, and b. Wang, "multi-modal feature fusion for spatial morphology analysis of traditional villages via hierarchical graph neural networks," arxiv preprint arxiv:2510.27208, 2025.