International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 3, May-Jun-2026, ISSN (Online): 2395-566X

VIBE SHIELD - Agentic Evolving Guard Intelligence System (AEGIS) for Wireless Networks

R Gayathri[1], Rohith V[2], M Mugilvannan[3]
[1]Assistant Professor Department of Computer Science Engineering
[2]UG Scholar, AI&DS Department
[1,2,3]Sri Venkateswara College of Engineering, Sriperumbudur TK, Kancheepuram DT, Tamil Nadu, India – 602117

Abstract- Sophisticated assailants outperform human defenders in today's cyber networks. This project introduces AEGIS, an end-to-end autonomous cyber operations system that integrates Large Language Models (LLMs) with Multi-Agent Deep Reinforcement Learning (MADRL) within the CybORG++ environment to overcome human latency and inflexible rule-based systems. AEGIS competes in a zero-sum game between an autonomous Blue Agent defense (Microsoft Phi-3.5-mini) and a Red Agent attacker (Qwen2.5-Coder-3B) using an Independent Learners technique under Decentralized Training and Decentralized Execution (DTDE). The system has a fully integrated 7-level progressive training pipeline with threshold-separated ChromaDB episodic memories, prioritized replay buffers, and LLM-specific action masking to remove hallucinations. The system uses a distributed MARL architecture that performs LoRA fine-tuning over two physical nodes via direct Ethernet, guaranteeing total parameter isolation, in order to maximize performance under stringent hardware restrictions. In the end, this architecture effectively illustrates how LLM agents with curriculum learning and episodic memory can independently learn intricate, multi-subnet cyber security tactics in sophisticated simulated environments.

Keywords – Multi-Agent Reinforcement Learning, Large Language Models, Autonomous Cyber Defense, Decentralized Execution, Action Masking, Parameter-Efficient Fine-Tuning, Zero-Sum Markov Games.

I. INTRODUCTION

A significant change in network security paradigms is required due to the critical operational mismatch between AI-driven, machine-speed cyberattacks and human-constrained, rule-based business defenses [2], [8]. The large, high-dimensional state-action spaces of dynamic corporate networks are difficult for traditional automated defensive systems, such typical Reinforcement Learning (RL) models, to handle because they lack intrinsic semantic reasoning. These models result in significant sampling inefficiency since they only map numerical states without having a fundamental understanding of cyber topologies or exploit mechanics.

On the other hand, whereas locally hosted Large Language Models (LLMs) have a thorough understanding of networking protocols, they are natively static text generators that cannot interact with real-world surroundings or retain tactical memory outside of token windows. Moreover, classic centralized Multi-Agent Reinforcement Learning (MARL) topologies run the possibility of hidden-state data leaking that compromises genuine zero-sum adversarial realism, and deploying these probabilistic LLMs in deterministic simulators often leads to invalid command hallucinations.
This study introduces the Agentic Evolving Guard Intelligence System (AEGIS) for Wireless Networks, theoretically known as VIBE SHIELD, to directly address these operational obstacles [1], [6]. This framework integrates fully localized LLMs with a Multi-Agent Deep Reinforcement Learning (MADRL) loop inside the CybORG++ simulator. It was designed as a system-wide integrated product rather than a straightforward API-based utility. AEGIS ensures that intelligent agents can constantly co-evolve by incorporating semantic reasoning and zero-shot deduction into the decision-making process through the embedding of LLMs as the active policy networks.

The main goal of this work is to show theoretically that locally hosted, hardware-constrained LLMs can operate as independent cyber agents that can perform dynamic, multi-step actions without the need for static heuristics. The following fundamental technical contributions enable this:

- **Decentralized Zero-Leakage Topology**: Creating a TCP-bridged, physically dispersed Decentralized Training and Decentralized Execution (DTDE) environment that rigorously enforces an Independent Learners (IL) methodology for true adversarial realism and mathematically prevents hidden-state data leakage.
- Deterministic Action Masking: Putting into practice a dynamic action-masking protocol that essentially connects

stiff simulator APIs with probabilistic natural language synthesis, totally eliminating invalid command hallucinations.

- Semantic Memory and Reward Shaping: By integrating a ChromaDB-backed statistical pattern extraction module with an independent Blue Coach to introduce dense, dynamic reward shaping, it is possible to counteract the context saturation and reward sparsity present in lengthy kill chains.
- Hardware-Optimized Fine-Tuning: Providing a sequential, memory-optimized Low-Rank Adaptation (LoRA) continuous training pipeline that rigorously adheres to consumer-grade hardware constraints, therefore democratizing multi-agent policy learning.
- Proactive Strategic Validation: Developing a tri-modal strategic assessment framework that experimentally demonstrates the agents' capacity to go beyond immediate reactionary measures and methodically create proactive, anticipatory defense mechanisms across a dynamically scaled curriculum is known as proactive strategic validation.

II. PREVIOUS WORKS

A. Review of Existing Works and System Limitations

Finite state machine simulators have played a major role in the development of autonomous cyber operations. The Cyber Operations Research Gym (CybORG) was developed by Standen et al. [13] to simulate enterprise network topologies; nevertheless, its initial implementations mostly used rigid, conventional neural networks for policy approximation. Using CybORG++, Emerson et al. [3] improved this infrastructure by incorporating PettingZoo multi-agent API support and improving observation techniques to allow competing models to be trained simultaneously. Wiebe and Al Mallah [14] successfully showed cooperative multi-agent reinforcement learning utilizing Proximal Policy Optimization (PPO) against a statically coded attacker by employing this improved topology.

Even with these fundamental developments, current automated defense systems have serious structural flaws. Standard neural networks (MLPs), which are the foundation of traditional Multi-Agent Reinforcement Learning (MARL) architectures, are intrinsically devoid of the episodic memory and semantic reasoning needed for intricate, multi-step threat interpretation. As a result, models are still susceptible to non-stationarity in the environment and are unable to dynamically adjust to new zero-day attacks. Additionally, cooperative defense against static, scripted opponents has been the main focus of previous research, which has not been evaluated in real zero-sum contexts where both agents actively co-evolve.

B. The Research Gap

The use of Large Language Models (LLMs) to close these intelligence gaps is now seriously out of alignment. Although the use of LLMs in cybersecurity is growing, their functions have been limited to passive advisory or single-agent offensive penetration testing (e.g., PentestGPT) [4]. Generative models have not been effectively implemented by current architectures as active, autonomous decision engines in a continuous, competitive MARL training loop [10], [12].

Table I compares the suggested technique with the most closely related previous work from the RMC-CMR research to clearly show the architectural leap that the AEGIS framework represents functioning as a system-wide integrated product using local models rather than a simple reactive API.

Table I. Architectural Comparison of MARL Cyber Defense Systems

Feature	RMC-CMR	AEGIS (Proposed System)
Architecture & Training	MLP optimized via PPO	Local LLMs (Phi-3.5, Qwen2.5) via LoRA
Adversarial Setup & Topology	Cooperative Blue vs. Scripted Red on a Single Node	Competitive Zero-Sum (LLM vs. LLM) via Distributed DTDE
Memory & Strategic Reasoning	Single-step reactive (No episodic memory)	3-Mode (Reactive/Preventive/Proactive) via ChromaDB
Reward Optimization	Standard sparse environment rewards	Dense dynamic coaching & action-based bonuses

III. PROPOSED METHODOLOGY

A. Game-Theoretic Formulation and Distributed Architecture

The multi-agent cyber combat is formalized by the AEGIS framework as a zero-sum, partially observable Markov game [9], which is defined by the tuple (1)

$$\mathcal{M} = \langle N, S, A, P, R, \Omega, O, \gamma \rangle, \qquad (1)$$

where N stands for the agents, S for the actual global state, A for the combined action space, P for transition probabilities, and R for the reward system. AEGIS replaces conventional mathematical policy approximations with the semantic reasoning of localized LLMs (Phi-3.5 for defense, Qwen2.5-Coder for offensive) in order to overcome the reactive limitations of conventional Multi-Layer Perceptrons (MLPs).

The system uses an Independent Learners (IL) approach under a Decentralized Training and Decentralized Execution (DTDE) topology to ensure complete adversarial parameter isolation. The architecture is physically divided into two compute nodes, as shown in Fig. 1. The Blue Agent and the CybORG++ simulation engine are hosted on PC1 (192.168.100.1), whereas the Red Agent and its corresponding memory are hosted separately on PC2 (192.168.100.2).

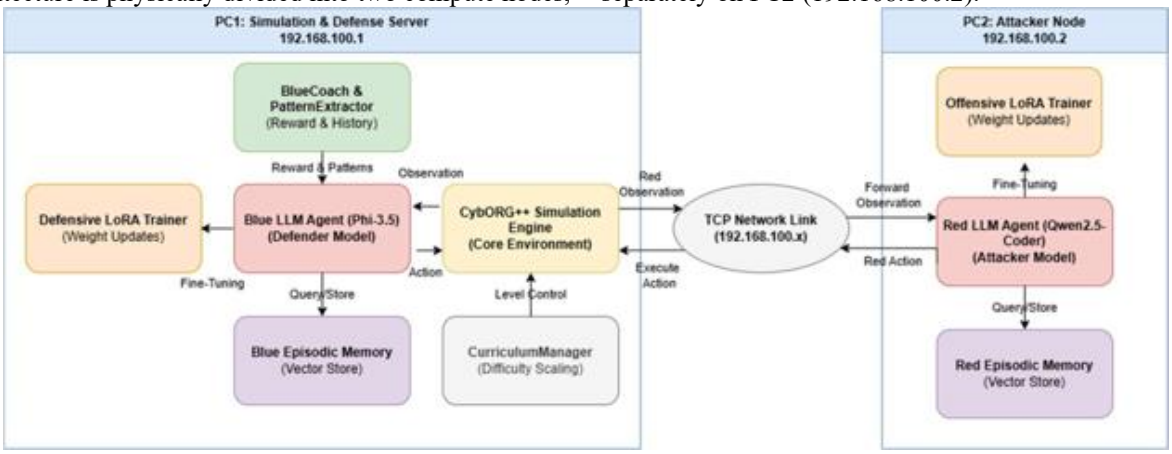

Fig. 1 AEGIS High-Level System Architecture

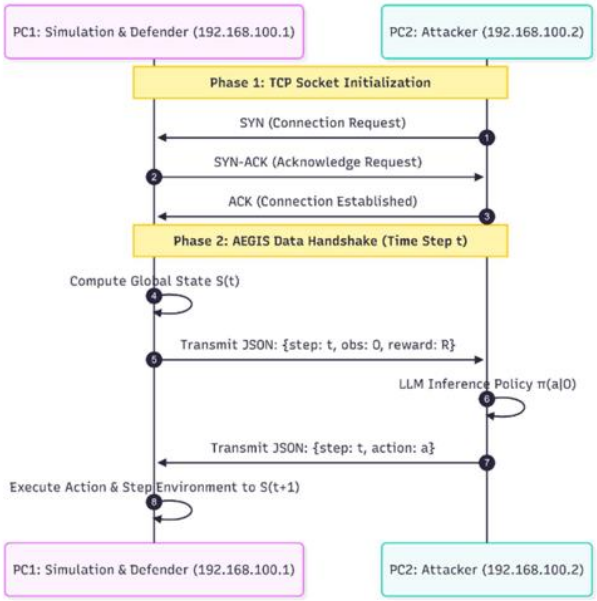

Fig. 2 Two-Phase TCP and AEGIS Handshake

The system uses a deterministic communication protocol to avoid temporal desynchronization during DTDE. A persistent TCP/IP socket, as shown in Fig. 2, uses explicit 4-byte payload framing to require a step-locked execution block, which stops the local simulation thread entirely until the adversarial integer action is received and decoded.

B. Agent Design: Multi-Modal Strategy and Phase Awareness

A 3-Mode Strategic Framework is integrated directly into the LLM context in order to circumvent simply reactive heuristics. This framework evaluates the state space S over concurrent temporal horizons to create a strategic context vector (2)

$$C_t = [M_{\text{react}}, M_{\text{prev}}, M_{\text{pro}}]. \quad \textbf{(2)}$$

This requires proactive forecasting of lateral movement(Mpro), preventive obfuscation (Mprev), and prompt remediation (M{react}). This is operationalized by mapping the discrete time step t and compromise state Scomp into an explicit

operational phase using a dynamic phase awareness function $\Phi(t, S\text{comp})$, which directs the policy $\pi(at|st, \Phi t)$ to dynamically shift as the kill chain matures (Table II).

Table II. Phase Awareness Mapping and Strategic Trigger States

Step (t)	Compromise State (Scomp)	Detected Phase (Φt)	Strategic Mode	Triggered Actions
1–5	User hosts clean	EARLY	Preventive	Misinform, Monitor
6–12	User hosts compromised	MID-EARLY	Reactive	Remove malware
13–18	Enterprise privileged	MID-LATE	Proactive	Preemptive Restore
≥ 19	Active Impact on Server	LATE	Emergency	Restore_ Op_Server0

C. Action Selection and Retrieval-Augmented Episodic Memory

An $\epsilon - greedy$ selector balanced by calibrated strategic biases W(a) is used to translate generative textual decisions into discrete environment actions. While basic surveillance is repressed (0.4 times), critical remedial operations, including restoring the operating server, are greatly multiplied (5.0 times). Every Ndecay=100 episodes, a historical decay algorithm (3) is run to avoid stale exploratory failures permanently distorting selection probabilities

$$C\text{new}(a) = \max(1, C\text{old}(a) \cdot \lambda)), \quad (3)$$

where $\lambda = 0.85$ acts as the decay factor.

A Statistical Pattern Extractor supported by ChromaDB returns quantitative viability data rather than raw text to prevent context saturation. To guarantee high-fidelity context, the system uses role-aware filtering (4)

$$M\text{filtered} = \{m \in M\text{db} | R\text{m} \geq R\text{min}\}. \quad (4)$$

Red employs Rmin=5.0 to prioritize successful exploits, while Blue uses Rmin = −100.0 to ignore catastrophic failures. Additionally, a retroactive distribution mechanism is used to solve offensive reward sparsity before storage (5)

$$r_{step} = \frac{R_{total}}{N_{recent}}. \quad (5)$$

D. Dynamic Reward Shaping and Strategic Coaching

The framework uses a dense, dynamic shaping mechanism to avoid sparse rewards in order to counteract the simulator's one-step observation latency. Equation (6) displays the total shaped reward at time t

$$R'_{blue}(s_t, a_t) = R_{env}(s_t, s_{t+1}) + B_{action}(a_t) + B_{coach}(H_t).$$

Baction provides instantaneous gradients upon command execution, while Bcoac assesses the historical window (Ht) to score behavioral timing and significant diversity (Table III).

Table III. Integrated Dynamic Reward Shaping Logic

Evaluated Condition / Action	Reward Type	Applied Value
Restore_Op_Server0 (Active Compromise)	Defense Bonus	8
Restore_Enterprise (Privileged State)	Defense Bonus	5
Anticipatory Preemptive Execution	Coach Bonus	3
Low Effort (Repetitive Monitoring Loops)	Coach Penalty	-0.3

A mathematical clipping function strictly scales the LoRA cross-entropy loss (7) as follows

$$\mathcal{L}_{final} = \mathcal{L}_{CE} \cdot \max\left(0.1, \min\left(2.0, \frac{R_{total}}{100.0}\right)\right), \quad (7)$$

to prevent dense rewards from causing catastrophic loss explosions within the underlying LLM neural architecture during fine-tuning.

E. Adaptive Curriculum and Hardware-Constrained Fine-Tuning

The 5-level adaptive curriculum used by the AEGIS pipeline is controlled by strict algorithmic fail-safes, such as a 25-episode demotion cooldown to stabilize regressing policies. Strictly constrained by a 6GB VRAM constraint, parameter-efficient LoRA upgrades [5] run sequentially every five episodes utilizing the Unsloth framework. If the rolling success of the Red agent exceeds a key threshold, its localized update is intentionally ignored in order to preserve the zero-sum equilibrium and allow the Blue defense computing cycles to recuperate(Eq. (8))

$$\text{Skip LoRA}_{red} \Leftrightarrow \frac{1}{100}\sum_{k=E-1}^{E-1} R_{red}^{(k)} > 5.0. \quad (8)$$

Lastly, during lengthy, unsupervised optimization cycles, an independent Policy Regulator enforces dynamic resource management by carrying out stringent PyTorch trash collection and averting memory allocation errors.

IV. EMPIRICAL EVALUATION AND DISCUSSION

A. Experimental Setup and Implementation Constraints Strict adversarial parameter isolation was ensured by deploying the AEGIS framework on a decentralized two-node hardware topology. Operating on Ubuntu Linux platforms with PyTorch support, both computational nodes needed a minimum of 8GB of VRAM in order to run Parameter-Efficient Fine-Tuning (PEFT) and host localized Large Language Models (LLMs) concurrently. The Blue Agent (Microsoft Phi-3.5) and the CybORG++ network simulation engine were housed on the primary server, whilst the Red Agent (Qwen2.5-Coder) was the only client on the secondary server. Ollama was used to controllocalized generative inference, and the Unsloth library was used to carry out sequential, continuous Low-Rank Adaptation(LoRA) updates that were controlled by programmatic resource swapping to avoid out-of-memory errors.

B. Quantitative Performance Metrics

Over the course of 72.14 hours, the framework successfully completed a huge continuous Multi-Agent Reinforcement Learning (MARL) loop, processing 1000 complete episodes at an average rate of 13.9 episodes per hour. Without halting, the automated Progressive Curriculum Learning module successfully handled the intricacy of the Markov Decision Process. The agents spent 12 episodes in Level 0, scaled through Level 1 (83 episodes), Level 2 (241 episodes), and Level 2.5 (214 episodes), before stabilizing in the Level 3 Full Scenario for 450 episodes, as the empirical breakdown illustrates. The environment's dynamic ability to keep models from overfitting to a static difficulty tier is mathematically demonstrated by the implementation of five curriculum advances and six demotions.

A significant level of LLM-driven autonomy and hardwareefficiency is shown by the action selection statistics obtained under a $\epsilon - greedy\ policy$ ($\epsilon = 0.05$). Table IV compiles the terminal output logs describing training stability and resource limitations.

Table IV. Agent Autonomy and Hardware Utilization Metrics

Metric	Blue Agent (Defender)	Red Agent (Attacker)
Generative LLM Model	Microsoft Phi-3.5-mini	Qwen2.5-Coder-3B
Total Action Selections	62,847	64,310
Valid Autonomous Actions	53,242	57,124
LLM Autonomy Rate	84.70%	88.80%
Unique Tactical Vectors	42	28
Peak VRAM Utilization	4.44 GB	3.97 GB
LoRA Convergence Loss	0.3589 (Step 600)	0.2618 (Step 597)

The quantized models effectively processed complicated, obfuscated JSON observation states with an average inference speed of 8200 ms per step, according to the real-time inference monitoring. Additionally, this continuous multi-agent reinforcement learning was implemented solely on consumer-grade hardware, effectively reducing model loss while closely adhering to the 6GB VRAM physical constraint.

C. Qualitative Strategic Analysis

Both agents went beyond random generation to create highly asymmetric, focused strategies, according to an analysis of the highest-frequency activities. Targeted lateral movement dominated the Red Agent's behavior; it cleverly determined that the Enterprise1 node was the best pivot point to breach the restricted Operational subnet, executing Privilege Escalation_User2 7,614 times and Exploit Remote Service_User2 8,732 times.

On the other hand, the network topology's structural weight was identified by the Blue Agent. The overall Blue action distribution strongly favored a twin strategy of decisive Remove (31.0%) and continual Monitor (37.0%), as seen in Fig. 3. As demonstrated by its absolute highest frequency action, Restore_Op_Server, which was conducted 9,847 times, the defending LLM learnt to prioritize the immediate restoration and defense of the crucial Operational server rather than waste compute cycles seeking the Red agent in the external User subnet.

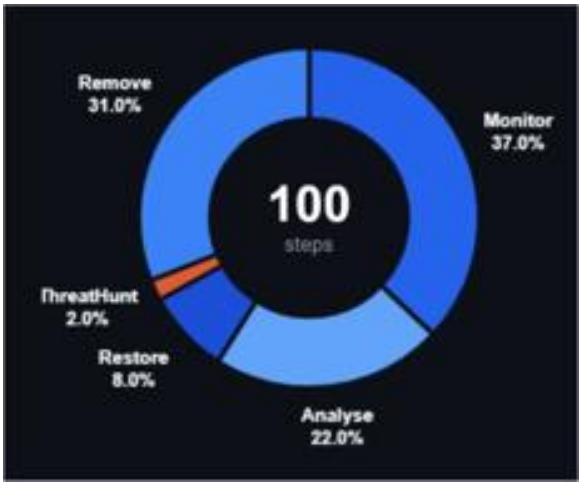

Fig. 3 Blue Action Distribution.

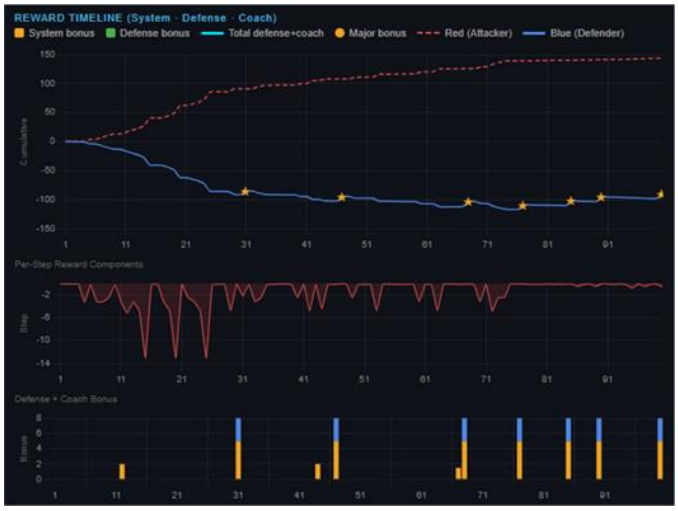

Fig. 4 Reward Timeline and Micro-Bonuses

Fig. 5 Real-Time Network Topology Security State

Furthermore, the reward timeline metrics confirm the effectiveness of the autonomous Blue Coach in neutralizing simulator observation delays. As demonstrated in Fig. 4, the Defense + Coach mechanism successfully injected micro-bonuses (e.g., +2, +5, +8) precisely at the moments the Blue agent executed valid preventative actions, ensuring the LLM received mathematical reinforcement for correct tactical choices even during periods of overall network compromise. Finally, the system translates the abstract CybORG++ environment into a strict visual network topology (Fig. 5). Because the Red agent's training artifacts remained physically isolated on PC2 until explicitly passed as obfuscated state variables, PC1 (Blue) and PC2 (Red) successfully synchronized their respective LoRA checkpoints and ChromaDB memory vectors at the conclusion of training sequences without implicit gradient leakage, successfully emulating a true zero-sum cyber engagement.

V. CONCLUSION AND FUTURE WORK

A. Conclusion

Overcoming the basic constraints of generative AI in deterministic environments is necessary to make the shift to autonomous cyber defense. By effectively combining Multi-Agent Deep Reinforcement Learning (MARL) and Large Language Models (LLMs), the AEGIS framework models enterprise cyberwarfare as a strict, zero-sum Markov Decision

Process. With no dependence on external API calls, the system operates as a fully localized, system-wide integrated product within a physically distributed Decentralized Training and Decentralized Execution (DTDE) architecture. The framework overcame several integration obstacles during its development:

- **Elimination of Hallucinations:** A novel Dual-Layer Action Masking protocol bridges probabilistic generation and discrete APIs, reducing invalid action hallucinations to 0% to ensure simulation stability.
- **Strategic Reasoning:** By using a 3-Mode Strategic Framework and an autonomous coaching module, adversarial agents are forced to go beyond basic reactive heuristics, effectively reducing reward sparsity.
- **Statistical Pattern Extraction:** By transforming episodic memory into a Statistical Pattern Extractor supported by ChromaDB, agents can dynamically filter and recover tactical vectors without going beyond token-window constraints.
- **Hardware Optimization**: The creation of a synchronized Low-Rank Adaptation (LoRA) pipeline demonstrates that continuous, multi-agent fine-tuning is possible on extremely limited hardware, rigidly adhering to a 6GB VRAM physical restriction.

B. Future Scope

The following crucial dimensions should be the focus of future study that builds upon this framework:

- **Real-Time Docker Emulation**: By switching from the abstraction CybORG++ simulator to live testing with Docker containers for stringent network isolation, autonomous agents may be tested against real network payloads and temporal latency.
- **Multi-Agent Topologies and Federated Defense**: To assess how several decentralized Blue LLMs (such as endpoint, firewall, and active directory agents) collaborate to outwit a coordinated Red team cluster [11], the existing 1v1 zero-sum architecture is expanded to a M vs. N topology.
- **Advanced Cyber Deception Techniques**: Adding active cyber deception to the Blue agent's action area, such as creating false network traffic to test adversarial reasoning or dynamically deploying containerized honeypots [7].
- **Adversarial Prompt Injection:** By enabling the Red agent to purposefully insert malicious payloads into simulated event logs, the framework's resilience is assessed. This leads to automated prompt injection attacks on the defender's LLM to investigate AI-on-AI exploitation.

REFERENCES

1. M. Abou Ali, F. Dornaika, and J. Charafeddine, "Agentic AI: A comprehensive survey of architectures, applications, and future directions," Artificial Intelligence Review, vol. 59, no. 11, 2026.
2. S. Baral, S. Saha, and A. Haque, "Autonomous cyber incident response using reasoning and action," in Proc. 2025 International Wireless Communications and Mobile Computing (IWCMC), 2025, pp. 1392–1397.
3. H. Emerson, L. Bates, C. Hicks, and V. Mavroudis, "CybORG++: An Enhanced Gym for the Development of Autonomous Cyber Agents," arXiv preprint arXiv:2410.16324v1, 2024.
4. A. Happe and J. Cito, "Getting pwn'd by AI: Pentration Testing with Large Language Models," in Proc. 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE), 2023, vol. 31, Art. no. 13371.
5. E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," in Int. Conf. on Learning Representations (ICLR), 2022, vol. 10, Art. no. 2106.09685.
6. K. Huang, Agentic AI: Theories and Practices, ser. Progress in Information Systems. Springer, 2025.
7. S. Inshi, R. Chowdhury, M. B. Taha, and C. Talhi, "Enhancing autonomy of context-aware self-healing in fog native environments," in Foundations and Practice of Security (FPS 2024), ser. Lecture Notes in Computer Science, vol. 15532. Springer, 2025.
8. N. Kshetri, "Transforming cybersecurity with agentic AI to combat emerging cyber threats," Telecommunications Policy, vol. 49, no. 6, Art. no. 102976, 2025.
9. Y. Kubera, P. Mathieu, and S. Picault, "IODA: An interaction-oriented approach for multi-agent based simulations," Autonomous Agents and Multi-Agent Systems, vol. 23, pp. 303–343, 2011.
10. J. Loevenich, E. Adler, T. Hürten, and R. R. F. Lopes, "Design and evaluation of an autonomous cyber defence agent using DRL and an augmented LLM," Computer Networks, vol. 262, Art. no. 111162, 2025.
11. Y. Ren, J. Wang, Z. Zhao, H. Wen, H. Li, and H. Zhu, "Automated tactics planning for cyber attack and defense based on large language model agents," Neural Networks, vol. 191, Art. no. 107842, 2025.
12. M. Rigaki, C. A. Catania, and S. García, "Building adaptive and transparent cyber agents with local language models," Expert Systems with Applications, vol. 299, Part A, Art. no. 129987, 2026.
13. M. Standen, M. Bowman, J. Richer, J. Gevers, and P. Parson, "CybORG: A Gym for the Development of Autonomous Cyber Agents," arXiv preprint arXiv:2108.09118, 2021.
14. J. Wiebe and R. Al Mallah, "Learning Cyber Defence Tactics from Scratch with Cooperative Multi-Agent Reinforcement Learning," Master's thesis, Royal Military College of Canada, 2023.