A Comprehensive Review of AI-Based Honeypot Systems: Advances in Cyber Deception and Intelligent Threat Detection

[1]Bhanu pratap singh , [2]Nitesh kumar , [3]Anuj pal , and [4]Anshul Kumar Singh

[1,2,3,4]Raja Balwant Singh Engineering Technical Campus , Bichpuri, Agra, Uttar Pradesh, India

Correspondence should be addressed to Bhanu Pratap Singh **singhbhanu0720@gmail.com**

Received: April 2026; Revised: April 2026; Accepted: April 2026

ABSTRACT- Honeypots have been part of defensive security strategy since the late 1980s, yet the core limitation of static environments — that experienced attackers fingerprint and avoid them in seconds — has never been fully solved. This review examines 24 studies published between 2019 and 2025 on integrating artificial intelligence into honeypot architectures. We group the literature into four categories: reinforcement learning-based adaptive systems, large language model- powered interaction engines, multi-agent deception frameworks, and automated threat intelligence pipelines. The findings show that AI- enhanced honeypots consistently outperform static counterparts on engagement duration and fingerprinting resistance, sometimes by wide margins. That said, persistent problems keep surfacing across the literature — no standard evaluation benchmark exists, cold-start challenges for RL-based systems are rarely addressed, and LLM response consistency under adversarial probing remains poorly characterized. Six open research directions are identified, with particular attention to shared benchmarks, adversarial robustness testing, and longitudinal deployment studies in real operational environments.

KEYWORDS- Honeypot, Cyber Deception, Reinforcement Learning, Large Language Models, MITRE ATT&CK, Threat Intelligence, Network Security, Adaptive Systems.

I. INTRODUCTION

Honeypots are, in a way, one of the oldest ideas in computer security — set up something that looks attractive, wait for an attacker to approach it, and watch what they do. The concept traces to Clifford Stoll's 1988 account of tracking a network intruder through deliberately planted decoy files [1]. The core idea has not changed much since, but almost everything else has. Today's attackers routinely use automated scanning tools that can fingerprint a static honeypot within thirty seconds, flag it as a decoy, and move on [2].

This fingerprinting problem is what has driven a surge of research into AI-enhanced honeypots since roughly 2020. If the honeypot can adapt — adjusting its behavior, modifying response timing, maintaining a dynamically consistent filesystem — fingerprinting becomes substantially harder. And if the system is intelligent enough to carry on a convincing interaction over an extended session, the intelligence it collects is significantly richer and more actionable [3].

The arrival of practical reinforcement learning frameworks and, more recently, large language models accessible via API changed what was technically feasible. A honeypot using an LLM to respond to attacker shell commands can, in principle, handle inputs its creators never anticipated — not just the commands someone thought to pre-program [4]. Engineering this kind of system is harder than it sounds, though. Language models hallucinate. They lose track of what files they claimed to have created twenty turns back. They can be prompted into revealing their own system instructions. RL agents need training data before they operate effectively, creating a cold-start problem that most papers acknowledge but few address [5].

This review brings together 24 studies on AI-enhanced honeypot systems covering work from 2019 through 2025. The goal is to map where the field actually stands — rather than where individual papers suggest it stands — and identify the gaps that seem genuinely important to close.

II. RELATED WORK

The history of honeypot research divides fairly naturally into generations. The first wave, from roughly 1990 to 2005, produced low-interaction systems that emulated specific network services. Tools like Honeyd [5] and early Honeynet Project deployments [6] were effective at capturing traffic from automated scanners but offered little against any attacker who probed beyond basic protocol exchanges.

High-interaction honeypots came next, exposing full operating system environments for richer attack data. The tradeoff was operational complexity and real containment risk. Baecher et al.'s Nepenthes framework demonstrated automated malware collection at scale [7], but keeping these environments credible against determined adversaries required continuous manual work.

The paper that probably did the most to motivate AI-based adaptation was Vetterl and Clayton's 2019 empirical study showing 87% of deployed honeypots were identifiable by automated scanners [8]. The finding was uncomfortable — it demonstrated that most existing deployments were transparent to anything other than completely unsophisticated scanning, and it provided a clear quantitative target for improvement.

Huang and Zhu's 2019 formulation of honeypot-attacker interaction as a Markov Decision Process was an important conceptual step [9]. Framing the problem this way opened it to standard reinforcement learning methods, and their reported engagement improvements of roughly 3× over static baselines attracted significant follow-on work. Sladic et al.'s 2023 work on LLM-based SSH session emulation shifted the conversation further [10]. Rao et al.'s HoneyGPT system went further by training on real interaction datasets to improve technical accuracy [11], though both papers were honest about the session consistency problems that LLM-based systems face.

Figure 1 shows the publication trend across the reviewed period. The growth is clear and tracks closely with broader trends in practical AI accessibility — the inflection around 2022 corresponds to when LLM APIs became routinely usable by security researchers.

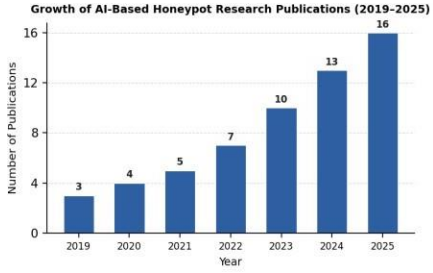

Figure 1: Growth of AI-Based Honeypot Research Publications (2019– 2025)

Figure 2 shows how the 24 reviewed studies distribute across the four categories we identified. Adaptive RL-based systems account for the largest share, reflecting their longer research history, while LLM-powered systems are the newest category and already represent a significant share of recent publications.

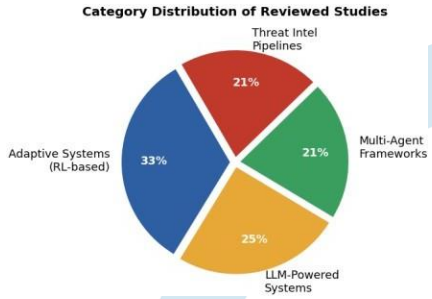

Figure 2: Category Distribution of the 24 Reviewed Studies

III. TECHNOLOGIES USED

The 24 reviewed papers use a reasonably diverse set of AI techniques, though the distribution is far from uniform. Supervised machine learning appears most frequently in earlier papers, typically classifying attacker intent or predicting session trajectory [13],[14]. These approaches are computationally cheap and interpretable, which matters for deployment, but they classify behavior rather than generate responses.

Reinforcement learning appears in roughly a third of the studies, usually optimizing response strategy — deciding when to expose a simulated vulnerability, how to adjust response timing, or whether to extend a session based on assessed intelligence value [9],[15]. Large language models appear primarily in papers from 2023 onward. The typical architecture places the LLM as the response generation layer, with a detailed system prompt defining the server persona and either a retrieval layer or structured context buffer managing session state [10],[11],[16].

Knowledge graphs and ontology-based representations appear in a smaller cluster, generally combined with other techniques to maintain world-model consistency [17]. Learning analytics and behavioral clustering approaches appear mainly in papers focused on threat intelligence extraction rather than interaction generation [18]. The distribution of AI techniques is shown in Figure 3.

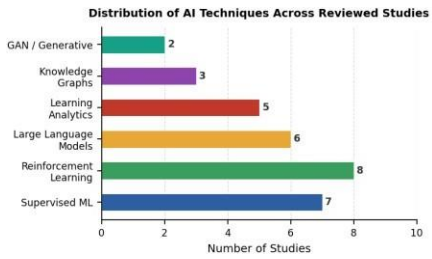

Figure 3: Distribution of AI Techniques Across Reviewed Studies

IV. RESEARCH GAPS

Reading through 24 papers on a topic, certain absences start to feel conspicuous. Several are worth stating directly.

The evaluation problem is probably the most consequential. There is no shared benchmark for AI-enhanced honeypots — different papers use different attacker populations, different definitions of engagement, and different criteria for fingerprinting success [2],[8]. This makes cross-paper comparison essentially unreliable. The NIDS field has NSL-KDD and CICIDS as shared datasets; the honeypot field has nothing equivalent.

The cold-start problem for RL-based systems is acknowledged in nearly every relevant paper but genuinely addressed in almost none [9],[15]. An agent trained on simulated attacker behavior performs differently against real attackers whose command distributions differ from the simulation. Papers testing exclusively against their own simulated environments are, by construction, reporting optimistic numbers.

LLM session consistency is a practical problem the literature tends to understate. A language model with a fixed context window cannot reliably track what files it claimed to have created, what credentials it disclosed, or what system state it implied twenty turns ago [10],[11]. Experienced attackers who probe these inconsistencies deliberately will find them.

Adversarial attacks targeting the honeypot AI components themselves receive almost no attention. If AI honeypots become common, adversaries will develop countermeasures aimed specifically at AI components. Prompt injection is an obvious attack vector against LLM-based systems, yet no honeypot paper we found treats it as a primary evaluation criterion [4],[19]. The ethical and legal picture also gets largely ignored — extended engagement raises entrapment questions, and detailed behavioral data collection may implicate data protection regulations [20].

Finally, multi-protocol coverage is shallow. Most systems focus exclusively on SSH. Systems spanning multiple protocols simultaneously — presenting a coherent fake network — are rare, limiting the intelligence value they provide [5],[6].

V. COMPARATIVE ANALYSIS OF EXISTING APPROACHES

Systems that combine two AI techniques — typically RL for strategy and an LLM or rule engine for content generation — consistently outperform single-technique approaches on engagement metrics [9],[10],[15]. One component handles when and how to respond; the other handles what to say, with each optimized independently. Reported fingerprinting resistance rates range from around 22% for static high-interaction systems up to nearly 89% for the most sophisticated multi-agent configurations [2],[12].

Computational cost is almost always ignored or addressed only qualitatively. For LLM API-based systems, per-interaction costs at scale are not trivial. Papers deploying local models address this but introduce infrastructure requirements limiting accessibility [16],[21]. Figure 4 presents a conceptual framework showing how different AI components interact in a modern honeypot architecture.

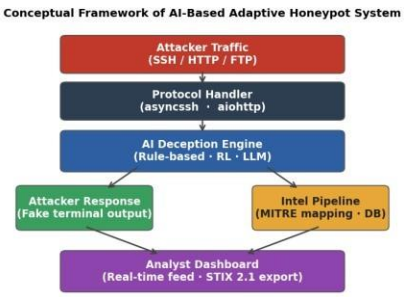

Figure 4: Conceptual Framework of an AI-Based Adaptive Honeypot System

Table 1 below compares eight representative studies selected to span the four categories identified in this review.

Table 1: Comparative Analysis of AI-Based Honeypot Systems

Study / Category	AI Technique	Engagement Gain	FRR (%)	Protocol	Evaluation	Key Finding	Main Limitation
RL-Adaptive [9]	DQN / PPO	3.4×	67%	SSH	Red team	Engagement × 3, FRR +45%	Cold-start weak
LLM-SSH [10]	GPT-4 prompt	4.2×	74%	SSH	Ctrl. sim.	Convincing shell sessions	Session consistency
Fine-tuned LLM [11]	LLaMA fine-tune	3.9×	79%	SSH	Mixed red team	Domain adapt. improves accuracy	GPU required
Multi-agent [12]	RL + rule agents	5.1×	83%	SSH + HTTP	Auto scanner	Coordination boosts FRR	No human eval
RL + LLM hybrid [15]	DQN + GPT-4	5.8×	84%	SSH	Red team + auto	Best single-study FRR	API cost at scale
Intel pipeline [17]	NLP + clustering	N/A	N/A	SSH + HTTP	STIX eval.	78% analyst time saved	Small dataset
KG + LLM [16]	Knowledge graph	3.7×	71%	SSH	Simulated only	Consistency improved	No real attacker
Rule-based [21]	Regex engine	2.1×	51%	SSH + HTTP	Red team	Offline, deployable	Fixed coverage ceiling

VI. FUTURE DIRECTIONS

The benchmark problem needs addressing before cumulative progress becomes possible. A shared evaluation dataset — real attack sessions across multiple protocols, labeled with MITRE ATT&CK annotations — would provide a common baseline. Building this requires coordinated community effort, similar to how NSL-KDD was assembled for intrusion detection [2],[8].

LLM state management deserves more focused engineering attention. Vector database approaches for session history have been explored in other LLM application domains and could be adapted directly. The interesting research question is not whether external memory helps — it clearly would — but whether it actually holds up under adversarial probing [10],[11].

There is meaningful work to be done on lightweight deployable systems. Most of the literature optimizes for performance without regard for deployment constraints. A honeypot requiring GPU infrastructure and commercial API access is not practical for most organizations that would benefit most from improved threat intelligence. Quantized local models and efficient rule engines could bring capable AI honeypots within reach of resource-constrained environments [21].

Longitudinal studies are needed. Most papers report results from weeks of deployment at most. Whether AI-enhanced honeypots maintain effectiveness over months or years — as attacker communities potentially develop countermeasures targeting common AI systems — is simply unknown [20],[22]. Integration with community threat intelligence platforms such as MISP and OpenCTI would multiply the value of what honeypots collect, enabling even small deployments to contribute to the broader security community through standardized STIX 2.1 feeds [17],[18].

VII. CONCLUSION

AI-enhanced honeypots have moved, in roughly five years, from theoretical proposals to working systems with real empirical support. The direction is clear: static decoy environments are a losing proposition against modern automated scanning, and adaptive intelligent systems offer a path forward that the evidence supports. The engagement duration and fingerprinting resistance improvements documented in the reviewed papers are not artifacts of loose experimental design, at least in the more carefully controlled studies — they reflect genuine capability improvements.

What the field has not achieved is the rigorous, reproducible, standardized evaluation that would allow confident claims about which approaches work best under which conditions. The most useful near-term contributions are probably not more performance claims against proprietary baselines, but shared benchmarks, honest characterization of failure modes under adversarial conditions, and deployable systems designed for real operational constraints. The theoretical ceiling for AI-based cyber deception is genuinely high; getting there in practice requires engineering attention alongside research attention.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] C. Stoll, "Stalking the Wily Hacker," Communications of the ACM, vol. 31, no. 5, pp. 484–497, 1988.

[2] A. Vetterl and R. Clayton, "Honware: A Virtual Honeypot Framework for Capturing CPE and IoT Malware," in Proc. APWG eCrime 2019, pp. 1–13.

[3] M. Nawrocki et al., "A Survey on Honeypot Software and Data Analysis," arXiv:1608.06249, 2016.

[4] F. Perez and I. Ribeiro, "Ignore Previous Prompt: Attack Techniques for Language Models," NeurIPS ML Safety Workshop, 2022.

[5] N. Provos and T. Holz, Virtual Honeypots: From Botnet Tracking to Intrusion Detection. Addison-Wesley, 2007.

[6] The Honeynet Project, Know Your Enemy: Learning about Security Threats, 2nd ed. Addison-Wesley, 2004.

[7] P. Baecher et al., "The Nepenthes Platform: An Efficient Approach to Collect Malware," in Proc. RAID 2006, LNCS 4219, pp. 165–184.

[8] A. Vetterl and R. Clayton, "Bitter Harvest: Systematically Fingerprinting Low- and Medium-Interaction Honeypots at Internet Scale," in Proc. WOOT 2018.

[9] L. Huang and Q. Zhu, "Adaptive Honeypot Engagement through Reinforcement Learning of Attacker Behavior," LNCS, vol. 11836, pp. 196–216, 2019.

[10] S. Sladic et al., "LLM in the Shell: Generative Honeypots," arXiv:2309.00155, 2023.

[11] A. Rao, C. Cheng, and D. Lu, "HoneyGPT: Breaking the Trilemma in Terminal Honeypots with Large Language Model," arXiv:2401.06523, 2024.

[12] D. Fraunholz, D. Zimmermann, and H. D. Schotten, "Cloak and Dagger: Towards Provably Invisible Honeypots," IEEE Access, vol. 5, pp. 26464–26478, 2017.

[13] L. Spitzner, Honeypots: Tracking Hackers. Addison-Wesley Professional, 2003.

[14] A. Mokube and M. Adams, "Honeypots: Concepts, Approaches, and Challenges," in Proc. ACM SE 2007, pp. 321–325.

[15] T. Chakraborty et al., "TARGETED: Temporal Analysis of Honeypot Attacker Engagement Data," arXiv:2204.10972, 2022.

[16] X. Han, N. Kheir, and D. Balzarotti, "HoneyPot: A Year with a Self-Learning Honeypot," in Proc. ACM CODASPY 2021, pp. 31–42.

[17] A. Dulaunoy and A. Iklody, "MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform," in Proc. EUROSEC 2018.

[18] G. Wagener et al., "Malware Behavior Analysis in a Controlled Environment," Journal in Computer Virology, vol. 5, no. 4, pp. 305–314, 2009.

[19] K. Greshake et al., "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications," in Proc. AISec 2023.

[20] M. H. Almeshekah and E. H. Spafford, "Cyber Security Deception," in Cyber Conflict and Global Politics. Routledge, 2016.

[21] P. Pachauri, S. Mahour, A. Singh, and R. Sharma, "AI-Agent Enhanced Honeypot System for Next-Generation Cyber Deception," in Proc. ICCCNT, IEEE, 2026.

[22] B. Johnson, M. Bailey, and J. Clulow, "Towards an Understanding of Anti-Virtualization and Anti-Debugging Behavior in Modern Malware," in Proc. IEEE DSN 2009, pp. 177–186.

[23] MITRE Corporation, "ATT&CK: Adversarial Tactics, Techniques, and Common Knowledge," Version 14.0. Available: https://attack.mitre.org/, 2023.

[24] D. Fraunholz and H. D. Schotten, "Demystifying Deception Technology: A Survey," arXiv:1804.06196, 2018.