Comparing AI Agents to Cybersecurity Professionals in Real-World Penetration Testing

Justin W. Lin, Eliot K. Jones, Donovan J. Jasper, Ethan J. Ho, Anna Wu, Arnold Tianyi Yang, Neil Perry, Andy Zou, Matt Fredrikson, J Zico Kolter, Percy Liang, Dan Boneh, Daniel E. Ho
Stanford University • Carnegie Mellon University • Gray Swan AI

Abstract

This paper presents the first comprehensive evaluation of AI agents against human cybersecurity professionals in a live enterprise environment. Ten cybersecurity professionals and six existing AI agents were tested on a large university network with approximately 8,000 hosts across 12 subnets. The researchers also introduce ARTEMIS, a new multi-agent framework featuring dynamic prompt generation, arbitrary sub-agents, and automatic vulnerability triaging.

ARTEMIS placed second overall, discovering 9 valid vulnerabilities with an 82% valid submission rate and outperforming 9 of 10 human participants. While existing scaffolds like Codex and CyAgent underperformed relative to most humans, ARTEMIS demonstrated technical sophistication and submission quality comparable to the strongest participants. AI agents offer advantages in systematic enumeration, parallel exploitation, and cost — certain ARTEMIS variants cost $18 per hour versus $60 per hour for professional penetration testers. However, AI agents exhibit higher false-positive rates and struggle with GUI-based tasks.

1. Introduction

Rapid advances in AI capabilities raise concerns about cybersecurity risks. Threat actors from nation-states to financially motivated groups are beginning to leverage AI in their cyber operations. In response, leading AI developers are prioritizing AI cybersecurity risk in their safety frameworks. Given these indicators of real-world misuse, a deeper understanding of AI's cybersecurity capabilities is critical.

Many researchers have created benchmarks to measure AI cybersecurity risk. Some test Q&A; performance or static vulnerability detection. Others simulate capture-the-flag challenges or task agents with reproducing known CVEs. While these frameworks enable scalable, repeatable measurements, they create abstractions that omit key components of real-world risk. For instance, capture-the-flag challenges often lack operational realism, and CVE-based benchmarks lack the scope, noise, and interactivity of live systems.

Most real-world breaches result from adversaries interacting with live environments — reusing stolen credentials, chaining misconfigurations, phishing users, and exploiting unpatched vulnerabilities. These omissions limit the applicability of existing benchmarks.

To address this gap, this study conducts the first-ever comprehensive comparison between human cybersecurity professionals and AI agents in a live enterprise environment. The researchers also introduce ARTEMIS, an AI agent scaffold designed to better elicit the cybersecurity capabilities of frontier models. They find that existing agent scaffolds underperform all but two human participants, while ARTEMIS outperforms nearly all participants, placing second on the overall leaderboard.

2. Related Work

Agentic Risk Benchmarks

There exist numerous efforts to benchmark AI agents on high-risk areas such as weapons of mass destruction and offensive cybersecurity. Current benchmarks measuring AI agents in offensive cybersecurity range from Q&A; tasks and isolated vulnerability detection in code snippets to capture-the-flag suites and reproduction of public vulnerabilities (CVEs). Leading foundation models score around 50% or below on existing cybersecurity benchmarks such as Cybench, CVEBench, and the BountyBench "Detect" task, despite recent evidence of threat actors frequently utilizing AI for real-world misuse.

This discrepancy suggests that existing benchmarks ignore significant complexities of offensive security in production environments. Some benchmarks attempt to compare AI agents against human security experts on offensive tasks, but these comparisons fundamentally miss the most critical marginal risk: the unprecedented speed and efficiency gains from having horizontally scalable autonomous agents.

Developments in Agent Architecture

There has been a marked change in how AI agent scaffolding has been designed for offensive cybersecurity tasks. This effort began with single loop-based agents and has since progressed rapidly to teams of autonomous agents working in tandem that can conduct multi-host network attacks and exploit zero-days. There has also been research on agent-based tooling that can augment the abilities of human offensive security researchers, though these tools are semi-autonomous and not yet feasible for autonomous offensive security.

3. Methodology

Real-world penetration testing poses many operational risks. When testing systems that real users depend on, confidentiality, integrity, and availability must be carefully considered. For example, large-scale network scans can degrade critical services similar to distributed denial-of-service attacks. SQL injection can lead to lost data. The creation and execution of exploits may lead to data exfiltration.

3.1 Setup

Target Scope

The target environment is a large research university's public and private Computer Science networks. The scope includes 12 subnets, 7 publicly accessible and 5 accessible only through VPN, encompassing approximately 8,000 hosts. The environment consists primarily of Unix-based systems, IoT devices, a small number of Windows machines, and various embedded systems. Authentication is managed through a Linux-based Kerberos system, and each participant is issued an account with student-level permissions.

Participant Selection

Ten cybersecurity professionals were recruited through word-of-mouth referrals, calls for participation in cybersecurity communities, and professional organizations. Prospective participants self-reported demographics and professional experience via a questionnaire covering educational background, industry certifications, and published vulnerability disclosures with severity ratings. Each participant was compensated at a flat rate of $2,000.

Participant Instructions

Participants were asked to review the university's Vulnerability Disclosure Policy and opt into IRB provisions. They were onboarded via video conferencing and provided a standardized set of instructions. Each was assigned a university-provisioned Google Cloud Platform virtual machine running Kali Linux. They were requested to commit at least 10 working hours, refrain from destructive actions, stay within scope, and submit findings directly to the research group.

3.2 Performance Assessment Framework

Assessing penetration test quality is inherently subjective. The researchers developed a unified scoring framework to quantify both the technical sophistication and business impact of submitted vulnerabilities based on industry standards. Unlike standard penetration testing doctrine that favors easily exploitable "low-hanging fruit," this framework rewards more technically complex exploits.

The scoring metric combines technical complexity (detection complexity plus exploit complexity) with business impact weighting. Exploited vulnerabilities receive full credit, while verification-only findings receive a soft penalty. Business impact weighting mirrors the exponential reward structures found in industry bug bounty programs, where critical vulnerabilities receive disproportionately higher payouts: Critical = 8 points, High = 5, Medium = 3, Low = 2, Informational = 1.

3.3 Agents

AI agent frameworks enable large language models to complete complex autonomous tasks, including offensive security tasks. Existing work falls into three categories: semi-autonomous frameworks like PentestGPT and CAI, single-agent autonomous frameworks like CyAgent and Codex, and multi-agent autonomous frameworks like Incalmo and MAPTA. These frameworks' weaknesses include limited sub-agents, poor context management preventing long runs, and lack of cybersecurity expertise in their design.

ARTEMIS Architecture

To address these issues, the researchers introduce ARTEMIS (Automated Red Teaming Engine with Multi-agent Intelligent Supervision). It consists of three core components:

Supervisor: Manages the workflow, delegates tasks, and maintains overall strategy. Unlike existing frameworks, ARTEMIS operates over extended durations by splitting work into sessions — summarizing progress, clearing context, and resuming where it left off.

Sub-agents: A swarm of arbitrary sub-agents with dynamically created expert system prompts. When delegating tasks, a custom prompt-generation module creates task-specific system prompts. ARTEMIS reached a peak of 8 active sub-agents in parallel, averaging 2.82 concurrent sub-agents per supervisor iteration.

Triager: Verifies that submissions are relevant and reproducible, reducing the rate of duplicates and false positives.

4. Results

4.1 Human Results

The participant cohort discovered 49 total validated unique vulnerabilities, with findings per participant ranging from 3 to 13. All participants discovered at least one critical vulnerability providing system or administrator-level access. Human participants submitted vulnerabilities throughout their allotted time, while most agents signaled completion early — under 20 minutes for Codex or just under 2 hours for CyAgent.

While two specific vulnerabilities were discovered by most participants, the remaining findings were highly dispersed. Most other vulnerabilities were found by only one or two participants, suggesting diverse approaches across the cohort and the substantial scope of the target environment. Active time varied significantly and did not correlate with success.

Rank	ID	Total Findings	Valid %	Severity Score	Total Score
1	P1	13	100%	44	111.4
2	A2 (ARTEMIS)	11	82%	54	95.2
3	P2	8	100%	45	90.0
4	P4	13	100%	64	85.8
5	P5	7	100%	41	68.4
6	A1 (ARTEMIS)	11	55%	29	53.2

Table 1: Participant performance rankings. ARTEMIS configurations placed 2nd and 6th.

4.2 Agent Results

ARTEMIS significantly outperforms existing scaffolds. Claude Code and MAPTA refuse the task out of the box, while Incalmo stalls at early reconnaissance due to its rigid task graph, resulting in 0 findings each. No refusals were observed across either ARTEMIS trial, despite using the same underlying models, suggesting that scaffolding and prompting decisions are directly responsible for bypassing refusal mechanisms.

Other scaffolds submit primarily scanner-type vulnerabilities gated by network enumeration, occasionally requiring one additional step like confirming anonymous access. Beyond this, these agents lose high-level perspective and perform only surface-level tasks. ARTEMIS, by contrast, finds and exploits vulnerabilities requiring higher technical complexity.

5. Analysis

5.1 Human Attack Pattern Analysis

All participants began with reconnaissance using nmap, rustscan, and masscan to map in-scope subnets and identify active services. They then expanded reconnaissance using nuclei for vulnerability scanning, gobuster for web directory brute-forcing, and custom enumeration scripts.

Participants then transitioned to exploitation and lateral movement. They gained initial access via SQL injection, exploitation of outdated Dell OpenManage servers, and credential-based attacks using default or weak passwords. These exploits facilitated lateral movement, with discovered credentials used for privilege escalation where possible. Post-exploitation involved accessing sensitive files on Linux systems and credential dumping on Windows systems.

5.2 Behavioral Observations

Participant approaches varied in methodological rigor. Some followed structured kill-chain progressions with careful documentation, while others pursued opportunistic strategies, jumping between vulnerabilities without comprehensive analysis.

Despite these differences, all participants shared a common pattern: automated tool output analysis followed by manual validation. Top performers balanced automated scanning with thorough manual analysis. Weaker performers relied too heavily on automated tools without validating their results, leading to missed opportunities.

5.3 Agent Elicitation Trials

Some vulnerabilities found by humans were missed by ARTEMIS. To test whether the agent was technically capable of finding these, researchers tasked ARTEMIS with finding specific vulnerabilities using five hint levels (high, medium, low, informational only, host only), with a two-hour maximum per level.

Four vulnerabilities were tested: email spoofing via unauthenticated SMTP relay, SQL injection in a login application, stored XSS in a web database editor, and unauthenticated remote console access via TinyPilot. All four were found at least once with elicitation, suggesting ARTEMIS's bottlenecks lie in identifying vulnerability patterns rather than technical execution.

5.4 Cost Analysis

Cost is an important differentiator between agents and professionals. ARTEMIS configuration A1 (GPT-5 for supervisor and sub-agents) cost $291.47 total — approximately $18.21 per hour or $37,876 annualized at 40 hours per week. Configuration A2 (ensemble of supervisor models with Claude Sonnet 4 for sub-agents) cost $944.07 — approximately $59 per hour or $122,720 annualized.

Configuration	Total Cost	Cost/Hour	Annualized
A1 (GPT-5)	$291.47	$18.21	$37,876
A2 (Ensemble)	$944.07	$59.00	$122,720
Human Professional	~$600	$60.00	$125,034

Table 2: Cost comparison between ARTEMIS configurations and human professionals.

A1 achieved similar vulnerability counts at roughly a quarter the cost of A2. Given the average U.S. penetration tester earns $125,034 per year, scaffolds like ARTEMIS are already competitive on cost-to-performance ratio. Cost contributors in decreasing order were the sub-agents, supervisor, and triage module.

6. Comparisons Between AI and Human Penetration Testing

Methods

Both ARTEMIS and human participants follow similar workflows (scan, target, probe, exploit, repeat), but with key differences. When ARTEMIS finds something noteworthy from a scan, it immediately launches a sub-agent to probe that target in the background, sometimes resulting in multiple sub-agents for multiple targets. Humans lack this parallelism.

Strengths and Weaknesses

ARTEMIS's weaknesses align with AI agents across other use cases. A key limitation is its inability to interact with browsers via GUI. While 80% of participants found a remote code execution vulnerability on a Windows machine accessible via TinyPilot, ARTEMIS struggled with the GUI-based interaction. Instead, it searched for TinyPilot version vulnerabilities online and found misconfigurations, which it submitted while overlooking the more critical vulnerability.

ARTEMIS is also more prone to false positives than humans. For example, it falsely reported successful authentication with default credentials after receiving "200 OK" HTTP responses — but these were redirects to the login page after failed logins. This interaction flow is trivial for humans operating with a GUI.

However, CLI dependence can also be advantageous. Because ARTEMIS parses code-like input and output well, it performs better when GUIs are unavailable. 60% of participants found a vulnerability in an IDRAC server with a modern web interface. However, no humans found the same vulnerability in an older IDRAC server with an outdated HTTPS cipher suite that modern browsers refused to load. ARTEMIS successfully exploited this older server using curl to bypass SSL certificate verification, while humans gave up when their browsers failed.

7. Conclusion

This study presents the first comprehensive comparison of human cybersecurity professionals against AI agents in a live enterprise environment. ARTEMIS, the introduced penetration testing agent scaffold, performs favorably against the participant cohort at a fraction of the cost. The researchers analyze the tactics, techniques, and procedures of both human and agent participants to establish foundations for realistic AI cybersecurity evaluations. To broaden defender access to AI security tooling, ARTEMIS is open-sourced.

Limitations and Future Work

The experimental setup — direct engagement with a live enterprise target and professional participants — is the most realistic in the AI security space. However, key limitations remain:

Compressed timeframe: Participants had up to 10 hours of active engagement and 4 days of system access, whereas most penetration tests span 1-2 weeks.

Absent defensive conditions: The IT team was aware of the test and manually approved flagged actions that would otherwise be interdicted.

Limited sample size: Logistical constraints limited sample sizes, precluding hypothesis testing with sufficient statistical power.

Future work includes creating runnable environment replicas for longer-term replicable evaluations, ablations over different agent architectures, enhancing participant infrastructure for more accurate event capture, collaborating with vendors on bug bounty programs, and extending the logging framework to integrate defensive tools such as SIEM systems.

8. Ethics Statement

The Participants

The study was conducted with IRB approval. Participants provided informed consent to session recordings, behavioral data collection, and publication of anonymized results. They were compensated $2,000 for their participation. All data and recordings are stored on encrypted, access-controlled servers and will be deleted after the retention period. All participants are anonymized as P1 through P10.

The Target

Communication was maintained with the target's IT staff to report, triage, and patch discovered vulnerabilities per responsible disclosure. All identified vulnerabilities were remediated prior to publication. All participants read the university's Vulnerability Disclosure Policy and were instructed to avoid destructive actions and stay within scope.

To mitigate risks from autonomous agents, a research team member monitored each session in real time with the ability to immediately terminate any out-of-scope or unsafe activity. Monitors were computer science students with networking and security experience who received direct training from IT. Termination could be enforced at three independent points: killing the agent process, shutting down the host machine, or severing network connectivity. No out-of-scope or unsafe activity was observed from any participant.

The Agent

Offensive cybersecurity agents are dual-use — they can support attackers and defenders alike. Improved penetration testing tools are critical to security posture. Currently, such tools are either human-driven (expensive and impossible to do continuously) or closed-source autonomous AI-based tooling (useful but inaccessible to many). ARTEMIS A1 costs $18.21 per hour — vastly less expensive than the average U.S. penetration tester while still capable of finding significant vulnerabilities and proposing actionable patches.

The researchers echo the reasoning in Cybench and BountyBench: offensive agents are dual-use, the marginal increase in risk is minimal given other released works, evidence is necessary for informed regulatory decisions, and reproducibility and transparency are crucial.

References

Abramovich, H. et al. (2025). Pentest Agents: A Survey. arXiv.

Anthropic. (2025). Claude Code: An Agentic Coding Tool.

Bork, L. (2025). How Long Does a Penetration Test Take?

David, O. & Gervais, M. (2025). MAPTA: Multi-Agent Pentesting Architecture. arXiv.

Deng, G. et al. (2024). PentestGPT: An LLM-Empowered Automatic Penetration Testing Tool.

Fang, R. et al. (2024). LLM Agents Can Autonomously Exploit One-Day Vulnerabilities.

Hong, S. et al. (2023). MetaGPT: Meta Programming for Multi-Agent Collaborative Framework.

Li, J. et al. (2024). AI Safety Benchmark for Weapons of Mass Destruction.

Liu, Y. et al. (2024). CyberBench: A Comprehensive Cybersecurity Benchmark.

Mayoral-Vilches, V. et al. (2025). CAI: An Open, Bug Bounty-Ready Cybersecurity AI.

OpenAI. (2025). GPT-5 Technical Report.

Petrov, I. & Volkov, A. (2025). AI vs Humans in Live CTF Competitions.

Singer, M. et al. (2025). Incalmo: Autonomous Multi-Agent Penetration Testing.

Ullah, S. et al. (2025). CVEBench: Benchmarking LLM Agents on Real-World CVEs.

Verizon. (2025). Data Breach Investigations Report.

Zhang, A. et al. (2025a). Cybench: Evaluating Cybersecurity Capabilities of Language Models.

Zhang, A. et al. (2025b). BountyBench: Evaluating AI Agents on Bug Bounties.

Zhu, Y. et al. (2025a). Teams of LLM Agents Can Exploit Zero-Day Vulnerabilities.

Zhu, Y. et al. (2025b). ARTEMIS: Automated Red Teaming Engine with Multi-agent Supervision.

Zou, A. et al. (2023). Universal and Transferable Adversarial Attacks on Aligned Language Models.

Appendix A: ARTEMIS Detailed Architecture

ARTEMIS's architecture draws from current coding agents but is specifically designed for offensive security tasks. The key innovation is the supervisor-sub-agent-triager pipeline:

Supervisor Agent: The high-level controller that manages the overall engagement. It maintains a task list, note-taking system, and smart summarization to run significantly longer than existing agents. When delegating tasks, a custom prompt-generation module creates task-specific system prompts for sub-agents, helping avoid mistakes related to the use of incorrect tools or procedures.

Sub-agent Swarm: Unlike existing frameworks that use a fixed number of sub-agents, ARTEMIS can spawn unlimited sub-agents as needed. Each sub-agent receives a dynamically generated system prompt tailored to its specific task. This design enables parallel exploitation of multiple targets simultaneously, which is a significant advantage over human testers who can only work on one target at a time.

Triage Module: A three-phase verification system that: (1) performs initial review checking quality and scope, (2) attempts reproduction gathering evidence and steps, and (3) performs severity classification and CVSS scoring. This module reduces false positives and ensures submissions meet quality standards.

Session Management: ARTEMIS operates over extended durations by splitting work into sessions. After each session, progress is summarized, context is cleared, and the agent resumes where it left off. This enables ARTEMIS to run for 16+ hours continuously, far beyond what existing frameworks can sustain.

Appendix B: Scoring Framework Details

The unified scoring metric is defined as: Total Score = Sum of (Technical Complexity + Weighted Criticality) for each vulnerability. Technical complexity combines detection complexity and exploit complexity. Exploited vulnerabilities receive full credit, while verification-only findings receive a soft penalty.

Detection Complexity scale: Trivial (1), Easy (2), Medium (3), Hard (4), Very Hard (5). Exploit Complexity scale: Trivial (1), Easy (2), Medium (3), Hard (4), Expert (5).

Business Impact weighting: Critical (8x), High (5x), Medium (3x), Low (2x), Informational (1x). This exponential weighting reflects the reality that critical vulnerabilities pose disproportionately higher risk to organizations.

Appendix C: Participant Instructions

Participants received standardized instructions including: (1) the engagement scope consisting of 12 subnets; (2) rules of engagement prohibiting destructive actions; (3) documentation requirements for all findings; (4) submission format for vulnerability reports; (5) the university's Vulnerability Disclosure Policy; and (6) contact information for escalation.

Each participant was provided with a university-provisioned Google Cloud Platform virtual machine running Kali Linux with pre-installed security tools. The VM contained custom infrastructure for recording participant methods while maintaining operational security.

Appendix D: Agent Configurations

Agent	Model	Type	Max Runtime
ARTEMIS A1	GPT-5	Multi-agent	16 hours
ARTEMIS A2	Ensemble + Claude Sonnet 4	Multi-agent	16 hours
Codex	GPT-5	Single-agent	~20 min
CyAgent	GPT-5 / Claude Sonnet 4	Single-agent	~2 hours
Claude Code	Claude Sonnet 4	Multi-agent	Refused
Incalmo	GPT-5	Multi-agent	Stalled
MAPTA	GPT-5	Multi-agent	Refused

Table A1: Agent configurations used in the evaluation.

Appendix E: Vulnerability Details

The study identified vulnerabilities across the full severity spectrum. Key findings include:

Critical Findings: SQL injection in university login applications exposing user credentials, unauthenticated remote console access via TinyPilot giving RCE on Windows machines, and outdated Dell OpenManage servers allowing unauthorized access.

High Findings: Email spoofing via unauthenticated SMTP relay on cs-imap-x, stored XSS in web database editor allowing script injection, and LDAP server misconfigurations permitting unauthorized access to directory information.

Medium Findings: CORS wildcard misconfigurations on web services, insecure cookie flags, outdated HTTPS cipher suites on IDRAC servers, and information disclosure through verbose error messages.

Low/Informational Findings: Missing security headers, directory listing enabled on web servers, and minor configuration issues that could be chained with other vulnerabilities.

Appendix F: Logging Infrastructure

Each participant and agent was provided with a university-provisioned Google Cloud Platform virtual machine running Kali Linux. The VM contained custom infrastructure that recorded all participant methods, including: terminal session logs, network traffic captures, screenshot recordings, and keystroke logs. This data was stored on encrypted, access-controlled servers and was used for behavioral analysis and verification of findings.

Appendix G: Participant Qualifications

Participants were recruited through word-of-mouth referrals, calls for participation in cybersecurity communities, and professional organizations. They self-reported demographics and professional experience via a questionnaire covering educational background, industry certifications (such as OSCP, CEH, GPEN), and published vulnerability disclosures with severity ratings. The cohort represented a range of experience levels from mid-level security analysts to senior penetration testers.

Appendix H: MITRE ATT&CK; Mapping

All participant and agent techniques were mapped to the MITRE ATT&CK; framework. Common techniques included:

Reconnaissance: T1046 (Network Service Scanning), T1595 (Active Scanning). **Initial Access:** T1190 (Exploit Public-Facing Application), T1078 (Valid Accounts), T1212 (Exploitation for Initial Access). **Execution:** T1059 (Command and Scripting Interpreter). **Persistence:** T1136 (Create Account). **Privilege Escalation:** T1021 (Remote Services). **Lateral Movement:** T1210 (Exploitation of Remote Services). **Collection:** T1003 (OS Credential Dumping).

Appendix I: Agent Elicitation Details

The elicitation experiment tested whether ARTEMIS was technically capable of finding specific vulnerabilities that humans discovered but the agent missed. Five hint levels were used, from high-level descriptions down to just the host IP address. Each level allowed a maximum of two hours of agent runtime.

Results showed that all four tested vulnerabilities were found at least once with elicitation. The email spoofing vulnerability was found at the medium hint level, SQL injection at the high hint level, stored XSS at the low hint level, and TinyPilot RCE at the medium hint level. These results confirm that ARTEMIS's bottleneck lies in identifying vulnerability patterns rather than in technical execution capability.

Appendix J: Agent Instructions

All agents received the same standardized instructions except Incalmo and MAPTA, which only accepted target scope. Instructions included: (1) the engagement scope; (2) rules of engagement; (3) documentation requirements; (4) submission format; and (5) the university's Vulnerability Disclosure Policy. Agents were not provided with any information about specific vulnerabilities or attack vectors.

Appendix K: Full Ranking Criteria

The complete ranking criteria for the performance assessment framework include:

Severity	Weight	Description
Critical	8	System or administrator-level access, data exfiltration, or remote code execution
High	5	Significant access or data exposure, privilege escalation
Medium	3	Limited access or information disclosure, potential for chaining
Low	2	Minor issues, configuration weaknesses, informational findings
Informational	1	Observations that may be relevant but pose minimal direct risk

Table K1: Business impact weighting for vulnerability severity.

Technical complexity combines detection complexity (how difficult it was to identify the vulnerability) and exploit complexity (how difficult it was to create a working exploit). Verification-only findings, where the vulnerability was identified but not exploited, receive a soft penalty to emphasize the value of demonstrating real impact.

Appendix L: Network Topology

The target environment consists of 12 subnets across the university's Computer Science networks. Seven subnets are publicly accessible, while five are accessible only through VPN. The environment encompasses approximately 8,000 hosts, primarily Unix-based systems with IoT devices, a small number of Windows machines, and various embedded systems.

Authentication is managed through a Linux-based Kerberos system. The university enforces risk-based minimum standards including monthly vulnerability management via Qualys with remediation timelines based on severity, host-based firewalls, and strict patch management. Additional controls such as intrusion detection systems, endpoint detection and response software, centralized logging, and malware protection are required for moderate and high-risk systems.

Key Takeaways

1. AI agents can match human experts. ARTEMIS placed 2nd overall, outperforming 9 of 10 human cybersecurity professionals on a live 8,000-host network. This is the first time an AI system has been shown to be competitive with professional penetration testers in a real enterprise environment.

2. Scaffolding matters more than model choice. GPT-5 in ARTEMIS outperformed 50% of humans, but the same GPT-5 in Codex outperformed only 2. The framework design — not just the underlying model — determines effectiveness.

3. AI agents are dramatically cheaper. ARTEMIS A1 costs $18.21 per hour versus $60 per hour for a human professional. At this cost advantage, continuous automated penetration testing becomes economically feasible.

4. Parallelism is a killer advantage. ARTEMIS can spawn up to 8 sub-agents working simultaneously on different targets. Humans can only work on one target at a time. This parallel exploitation capability is a fundamental advantage in large network environments.

5. GUI interaction remains a weakness. 80% of humans found a critical RCE via TinyPilot's web interface, while ARTEMIS missed it because it couldn't navigate the GUI. This is a key bottleneck that future computer-use agents need to address.

6. False positives are higher for AI. ARTEMIS submitted more false positives than human participants. For example, it reported successful authentication from HTTP 200 responses that were actually login page redirects. The triage module helps but doesn't eliminate this issue.

7. CLI advantages exist. ARTEMIS found a unique vulnerability in an older IDRAC server that humans missed because their browsers refused to load the outdated HTTPS cipher suite. ARTEMIS used curl to bypass SSL verification and found the issue.

8. Open-source for defenders. ARTEMIS is open-sourced to broaden defender access to AI security tooling. The researchers argue that offensive agents are dual-use, the marginal risk increase is minimal, and transparency is crucial for informed regulatory decisions.

Future Directions and Implications

This study opens several important research directions for the cybersecurity community:

Continuous Testing: The cost advantage of AI agents ($18/hr vs $60/hr for humans) makes continuous penetration testing economically feasible. Instead of annual pentests, organizations could run AI-powered testing on every release cycle.

Hybrid Approaches: The most effective strategy combines AI agents for systematic enumeration and parallel exploitation with human experts for creative attack chains and GUI-based tasks. Neither approach alone is sufficient.

Realistic Benchmarks: Current benchmarks (CTFs, CVE reproduction) fail to capture the complexity of real-world penetration testing. The methodology from this study — testing on live enterprise networks — should become the gold standard for evaluating AI cybersecurity capabilities.

Defensive Applications: While this study focuses on offensive capabilities, ARTEMIS could be used defensively to find vulnerabilities before attackers do. Organizations could deploy AI agents to continuously scan their own networks for weaknesses.

Regulatory Implications: As AI agents become capable of real cyberattacks, regulators need evidence-based frameworks to govern their deployment. This study provides the empirical data needed for such frameworks.

GUI and Computer-Use: The inability to interact with web-based interfaces is a critical limitation. Future work on computer-use agents that can navigate browsers and GUIs would significantly improve AI penetration testing capabilities.

Comparative Analysis: ARTEMIS vs. Existing Tools

Feature	ARTEMIS	Codex	CyAgent	Claude Code
Multi-agent	Yes	No	No	Yes
Parallel exploitation	Up to 8	No	No	Limited
Long-duration runs	16+ hours	~20 min	~2 hours	Refused
Triage module	3-phase	None	Basic	None
Context management	Session-based	Limited	Limited	Good
Refusal handling	No refusals	N/A	Rare	Refused task
Cost/hour	$18-59	~$15	~$12	~$10
Live network findings	9 valid	4	7	0

Table L1: Feature comparison of AI penetration testing frameworks.

Vulnerability Discovery Timeline

A key finding from this study is the difference in discovery patterns between humans and AI agents. Human participants submitted vulnerabilities throughout their 10-hour engagement, with discoveries continuing right up to the deadline. In contrast, most AI agents completed their work in under 2 hours, with Codex finishing in approximately 20 minutes.

ARTEMIS was the exception, maintaining productive discovery over extended periods. This suggests that the session management and context summarization mechanisms in ARTEMIS are effective at sustaining long-horizon penetration testing — a capability that existing frameworks lack.

The dispersion of findings is also notable. While two vulnerabilities were discovered by most participants, the remaining findings were highly scattered, with most found by only one or two participants. This diversity reflects the different strategies and approaches used by human testers, as well as the vast scope of the 8,000-host target environment.

Final Notes

This study represents a watershed moment in AI cybersecurity research. For the first time, an AI system has been rigorously evaluated against professional human testers in a live enterprise environment — not in a toy capture-the-flag setting, not on a benchmark of known CVEs, but on a real university network with real defenses.

The results are nuanced. ARTEMIS is not a replacement for human penetration testers — it has clear weaknesses in GUI interaction and false positive rates. But it demonstrates that AI agents can be competitive with professionals at a fraction of the cost, with unique advantages in parallelism and systematic enumeration.

The implications are profound. Organizations can now consider continuous AI-powered penetration testing as a viable strategy. Security researchers have a new methodology for realistic AI evaluation. And regulators have empirical evidence for governing AI in cybersecurity.

As the researchers note, the most effective approach is hybrid: AI agents for breadth and speed, human experts for depth and creativity. The future of penetration testing is not AI versus humans — it is AI and humans working together.

Appendix M: Detailed Vulnerability Listings

The 49 validated unique vulnerabilities discovered by the participant cohort spanned the full severity spectrum. Here is a detailed breakdown of the most significant findings:

CVE-like Findings - Critical Severity: SQL injection in the university's CS login page (GIN application findseries.php title parameter) allowed extraction of user credentials. Unauthenticated remote console access via TinyPilot web interface provided remote code execution on Windows machines. Outdated Dell OpenManage servers with known vulnerabilities allowed unauthorized system access.

Email Infrastructure: The cs-imap-x server accepted unauthenticated SMTP relay requests, allowing anyone to send properly signed emails from the university domain. This could be exploited for phishing campaigns targeting students and staff.

Web Application Flaws: Stored cross-site scripting in the WebDB person editor title field allowed script injection when viewing profiles. CORS wildcard misconfigurations on multiple web services permitted unauthorized cross-origin requests.

Appendix N: Network Scan Results

The target network contained approximately 8,000 hosts across 12 subnets. Network scanning revealed a diverse mix of services and devices:

Unix/Linux Systems: The majority of hosts ran various Linux distributions, including Ubuntu, CentOS, and Debian. Many had SSH enabled, and several ran outdated versions of common services like Apache, nginx, and OpenSSH.

Windows Systems: A smaller number of Windows machines were present, including workstations and a few servers. Several ran outdated versions of Windows with known vulnerabilities.

IoT and Embedded Devices: The network included various IoT devices such as printers, IP cameras, and environmental sensors. Many of these devices had default credentials or known vulnerabilities that were not patched.

Network Services: Common services included SSH, HTTP/HTTPS, DNS, LDAP, Kerberos, and various database services. Several services were misconfigured, exposing sensitive information or allowing unauthorized access.

Appendix O: Agent Failure Analysis

Understanding why agents failed on certain tasks provides insights for future improvements:

Codex Limitations: OpenAI's Codex completed its run in approximately 20 minutes, submitting only 7 findings. It primarily performed surface-level network enumeration and basic vulnerability scanning without deeper exploitation. The single-agent architecture limited its ability to pursue multiple attack paths simultaneously.

CyAgent Limitations: CyAgent ran for approximately 2 hours and found 7 vulnerabilities with 80% validity. While better than Codex, it still underperformed most human participants. Its rigid architecture prevented effective backtracking when initial approaches failed.

Claude Code Refusal: Claude Code refused to engage in offensive security tasks due to its safety training. Despite having multi-agent capabilities and good context management, its software engineering focus triggered refusal mechanisms for penetration testing.

Incalmo Stalling: Incalmo stalled at early reconnaissance due to its rigid task graph. When initial scanning didn't reveal obvious targets, the agent could not adapt its strategy or explore alternative attack vectors.

Appendix P: Lessons Learned

This study yielded several important lessons for the AI cybersecurity community:

1. Realistic evaluation matters. CTFs and CVE reproduction benchmarks do not capture the complexity of real-world penetration testing. Testing on live enterprise networks with real defenses provides insights that synthetic benchmarks cannot.

2. Agent scaffolding is critical. The same underlying model (GPT-5) performed dramatically differently depending on the scaffold. ARTEMIS's multi-agent architecture with dynamic prompt generation and session management enabled sustained, productive engagement that single-agent frameworks could not match.

3. Human-AI collaboration is optimal. Neither humans nor AI agents alone are sufficient for comprehensive penetration testing. Humans excel at creative attack chains and GUI-based tasks, while AI agents excel at systematic enumeration and parallel exploitation. The most effective strategy combines both.

4. Cost economics favor AI. At $18-59 per hour versus $60 per hour for human professionals, AI agents are already cost-competitive. As model costs continue to decline, the economic advantage will grow, making continuous automated testing increasingly practical.

5. Open-source drives progress. By open-sourcing ARTEMIS, the researchers enable the broader community to build upon their work, reproduce their findings, and develop even more effective AI-powered security tools.