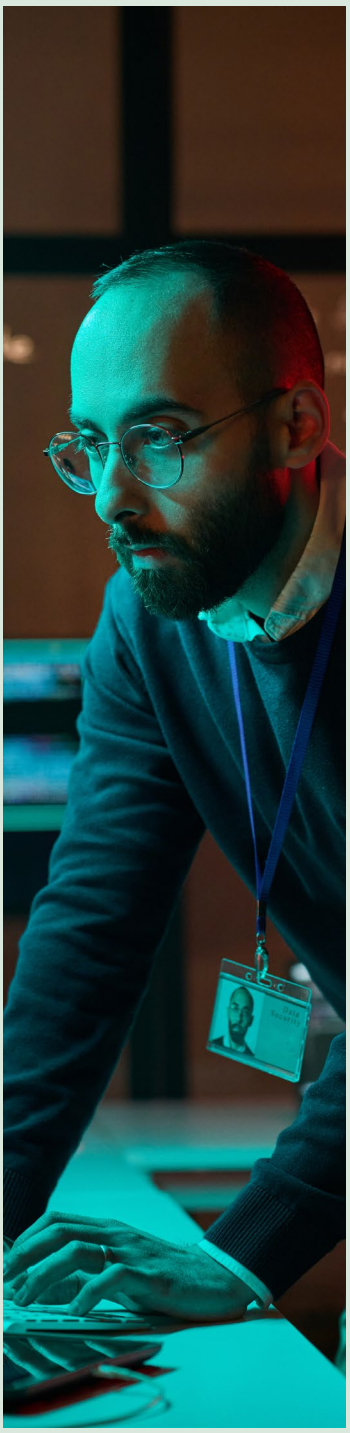
WHITE PAPER



MIRAGE: CLOSING THE MALWARE DRIFT GAP BEFORE IT BECOMES A BREACH

Infosys
Navigate your next

Executive Summary

Malware is dynamic. It keeps shifting around once a system to detect it is up and running. Defences lose ground if that change goes unnoticed for even a day.

This paper introduces MIRAGE, a three-part detection framework built by Infosys Cybersecurity researchers. The framework consists of:

- A multi-view Portable Executable (PE) file scanner that analyses files for malware in four independent ways at once (its raw contents, any embedded text, its internal structure, and what it actually does when run)
- A drift-detection layer that continuously watches for early signs of the scanner's accuracy slipping, and uses AI reasoning to work out why, rather than waiting for someone to notice weeks later
- A self-healing loop that automatically fixes the specific part of the scanner that's falling behind as malware evolves. It also checks that the fix actually worked, and puts the improved version back into service, with a named person required to approve the biggest changes.

In tests on industry-standard malware benchmarks such as BODMAS and EMBER, the MIRAGE framework caught previously unseen malware families with 97.3% accuracy at a very low false-positive rate, and cut the time to detect a new drift event by 94% as compared to typical daily monitoring. Every step in the process is auditable, and the biggest decisions still require a named person's sign-off.

The paper explains why detection needs to become a continuous loop rather than a one-time decision, and how the framework works at a level any security leader can follow.



External Document © 2026 Infosys Limited

A Dynamic Malware Detection System

Picture a new strain of malware appearing somewhere on a corporate network. The detection system in place was trained on last month's threats. By the time analysts notice the anomaly, confirm it, write a signature, and push the update, the malware has often already changed shape again.

For most enterprise security stacks today, this is the default operating condition.

For three decades, malware detection has largely followed the same pattern: a new threat appears, analysts study it, a signature gets written, and detection databases get updated. That model only works if new threats emerge more slowly than defenders can respond.

This is a reasonable assumption when malware authors are people working at human speed. It becomes a much riskier one as AI tools increasingly assist in discovering vulnerabilities and generating attack variants.

Claude Mythos Preview (Anthropic, 2026) demonstrated that frontier AI systems can autonomously identify thousands of previously unknown vulnerabilities across all major operating systems, complete full attack chains in under 30 minutes, and generate malware that looks like a completely different file every time it copies itself — different byte patterns, so a signature-based scanner sees no match to anything in its database — while still undertaking precisely the same malicious actions underneath. When attack generation operates at this cadence, signature-based defences lose their fundamental operating assumption: time.

Most enterprise security setups have three structural gaps.

First, traditional detection relies on a database of known threat fingerprints or 'signatures'. If new malware variants show up faster than that database can be updated, the system simply doesn't recognize them as a threat.

Second, many detection approaches only watch what a program does after it starts running, by tracking logs and behaviour. They have zero visibility into what the file itself looks like before it begins executing, relying solely on post-execution behavior. Third, even the accurate, modern AI-based scanners tend to work as black boxes. They spit out a verdict, malicious or not, but don't show any of their internal reasoning. That's a problem if a system must evaluate that reasoning afterward and analyse it to figure out why the scanner is starting to get things wrong, or to automatically improve it.

MIRAGE has been designed to close these gaps.

Rethinking Detection as a Continuous Loop

The core thesis of MIRAGE is that detection is an ongoing adaptation problem, rather than a one-time engineering decision. The system must detect threats, understand why its detection is degrading, fix the specific failing component, verify the repair, and update the live scanner - autonomously, at machine speed.

A machine learning model that's accurate on day one will, left alone, become steadily less accurate as the threats it faces continue to change, a phenomenon researchers call concept drift. Rather than treating that decline as something to notice weeks or months later during a scheduled retrain, MIRAGE:

01 Detects its own degradation as it happens

02 Works out which specific part of its judgement is failing and why

03 Fixes that part rather than retraining everything from scratch

04 Verifies that the fix actually helped, and

05 Puts the improved version back into live service, largely without a human needing to intervene for anything short of the biggest decisions.

External Document © 2026 Infosys Limited

The contributions of this paper are:

- A multi-view cross-attention PE scanner producing calibrated confidence, per-view attribution weights, and a 640-d embedding designed for temporal drift analysis - not only classification.
- A dual-track drift detection pipeline combining batch statistical monitoring with per-sample streaming anomaly detection achieving sub-200ms latency.
- An LLM-augmented agentic adaptation loop with selective LoRA encoder retraining, adversarial guard layers, and a five-tier autonomous action ladder with formal human-in-the-loop boundaries.
- Formalisation and empirical evaluation of Exponential Vulnerability Operations: a time-decayed risk scoring model with measurable SLA compliance properties under AI-speed attack cadence.

How It Works

The Scanner: Multi-view Cross-attention Scanner

The MIRAGE multi-view cross attention scanner analyses each Portable Executable (PE) file through four architecturally independent view encoders, simultaneously.

- One view studies the file's raw byte-level patterns.
- A second looks at the readable text and identifiers embedded in the file.
- A third examines the file's internal structure: how it's laid out, where its entry points are, and whether that layout looks tampered with.
- A fourth maps out which system functions the file calls and how they connect, which is often the clearest signal of what a file is actually trying to do.

MIRAGE weights them the four views dynamically depending on the situation. A file that looks deliberately disguised gets judged more heavily on its byte patterns; a file that looks like it's piggybacking on a trusted piece of software gets judged more heavily on what it calls. The result is a single, calibrated confidence score, along with a record of which of the four views drove that particular judgement, useful both for human review, and for the next stage of the loop.

The Watcher: Drift Detection and Risk Scoring

Behind the scanner, a second layer continuously tracks how its judgements change over time, on two tracks running in parallel.

Track A watches broad statistical patterns across batches of files, looking for the kind of gradual shift that signals a new malware family emerging.

Track B checks every single file as it arrives, in under a fifth of a second, for the kind of sharp, individual anomaly a batch check might not catch until much later.

When either track raises a flag, a dedicated reasoning layer (a locally deployed language model rather than a general-purpose chatbot) investigates why, working only from statistical summaries rather than raw file content. To guard against the reasoning layer reaching a confident sounding but wrong conclusion, it reasons through the same question twice, independently. The two answers are checked against each other before anything gets acted on.

Each confirmed finding is then turned into a single risk score that weighs severity, how far the threat could spread if unaddressed, how unusual the pattern is, and how likely it is to already be a known, exploited vulnerability.



External Document © 2026 Infosys Limited

The Healer: Agentic Adaptation Loop

The final agentic layer decides what to do about each flagged issue, on a five-step ladder that scales the response to the severity of the finding.

- NO_ACTION: A log entry only.
- ADJUST_THRESH: A fully automatic internal adjustment measured in milliseconds.
- FLAG_CLUSTER: A flag routed to a human analyst for labelling.
- SELECTIVE_RETRAIN: A tightly scoped retraining of the specific part of the scanner that is drifting. This uses a lightweight fine-tuning technique that updates a small slice of the model, instead of rebuilding from scratch. This always validates against held-out data before going live.
- FULL_RETRAIN: A full retrain. This waits in a pending state until a named analyst explicitly approves it.

After any change, the system checks whether the fix actually improved things before treating it as done. If it doesn't, the issue is escalated. The improved component is then loaded back into the live scanner, closing the loop.

Urgency That Compounds

Not every flagged issue is equally urgent, and MIRAGE doesn't treat them as static once raised. The longer a moderate-risk alert sits unaddressed, the more urgent the framework treats it, on a curve that steepens the higher the alert's starting severity. So, a middling-risk issue ignored for a day can become as pressing as a high-severity one addressed immediately.

Under a faster-moving attack scenario, the kind that security researchers increasingly model as AI tools shorten the time between vulnerability discovery and exploitation, that urgency curve compresses further, and response windows shrink accordingly.

This accelerated scenario is a stress test the research team constructed to pressure-test response times. It does not reflect the demonstrated capabilities of any specific, commercially available AI system.

Test Results

The framework was evaluated against two of the field's standard public malware research datasets: BODMAS, which provides over 134,000 labelled files across 581 malware families with monthly timestamps spanning August 2019 to September 2020 (making it possible to test how accuracy holds up over time, rather than only at a single point), and EMBER v2, a 1.1-million-sample benchmark widely used for pre-training and baseline comparison. A third dataset, SOREL-20M, was used to train the fast per-file anomaly check. All results are reproducible against public benchmarks.

On raw detection accuracy, the full system outperformed both a strong industry baseline and simpler versions of itself.

Method	Detection rate*	Overall accuracy (AUC)	Confidence reliability†
EMBER LightGBM (industry baseline)	92.9%	0.999	0.041
MalConv (prior published baseline)	89.4%	0.994	0.063
MIRAGE: single view only	93.8%	0.997	0.038
MIRAGE: without confidence calibration	96.1%	0.999	0.057
MIRAGE (full system)	97.3%	0.9995	0.012

**True positive rate at a 0.1% false-positive rate, the standard way to measure detection accuracy while keeping false alarms rare. †Expected Calibration Error; lower is better and measures how well the system's stated confidence matches its actual accuracy.*

External Document © 2026 Infosys Limited

The gap between the full system and its single-view counterpart (93.8% vs. 97.3%) confirms that combining the four perspectives isn't redundant: each one catches things the others miss. The gap in the confidence-reliability column is arguably more important: removing the calibration step barely changed raw accuracy, but made the system's stated confidence nearly five times less trustworthy. That matters because confidence scores feed directly into downstream automated decisions: a model that's often right but doesn't know when it's wrong is a much harder system to trust with autonomous action.

On drift detection specifically (how quickly the system notices its own performance slipping, and how well it recovers), results were tested by simulating seven new malware families that were absent during training:

Approach	Mean Time To Detection (MTTD)	Accuracy after recovery
No adaptation	Never: accuracy just degrades	74.2%
Monthly retraining	312 hours (~13 days)	91.3%
MIRAGE: batch monitoring only	18.4 hours	94.7%
MIRAGE (full system)	1.1 hours	97.1%

The full system detected drift 94% faster than a typical daily-monitoring baseline, and recovered to 97.1% accuracy within a single automated retraining cycle, without a human needing to diagnose the problem first. Under a simulated accelerated-attack scenario, the system's highest-severity alerts were flagged, and affected endpoints isolated within roughly four minutes on average, comfortably inside every target response-time threshold tested.

Limitations and Future Work

MIRAGE operates exclusively using static PE file analysis. It doesn't run them in a sandboxed environment to observe live behaviour. This means that heavily disguised files that resist static analysis are flagged for human review instead of being classified with confidence. Adding that dynamic layer is a natural next step.

The explanations of the reasoning layer are not yet formally verified. The built-in cross-check catches inconsistent reasoning, but it isn't a guaranteed defence against a determined attempt to manipulate the system's judgement. That is an area that needs more rigorous testing before wider deployment.

Finally, all of this evaluation uses public research datasets: validating the urgency-scoring timings against real security operations centre data, and building a proper benchmark for AI-generated malware specifically, remain open problems for the whole field, this framework included.



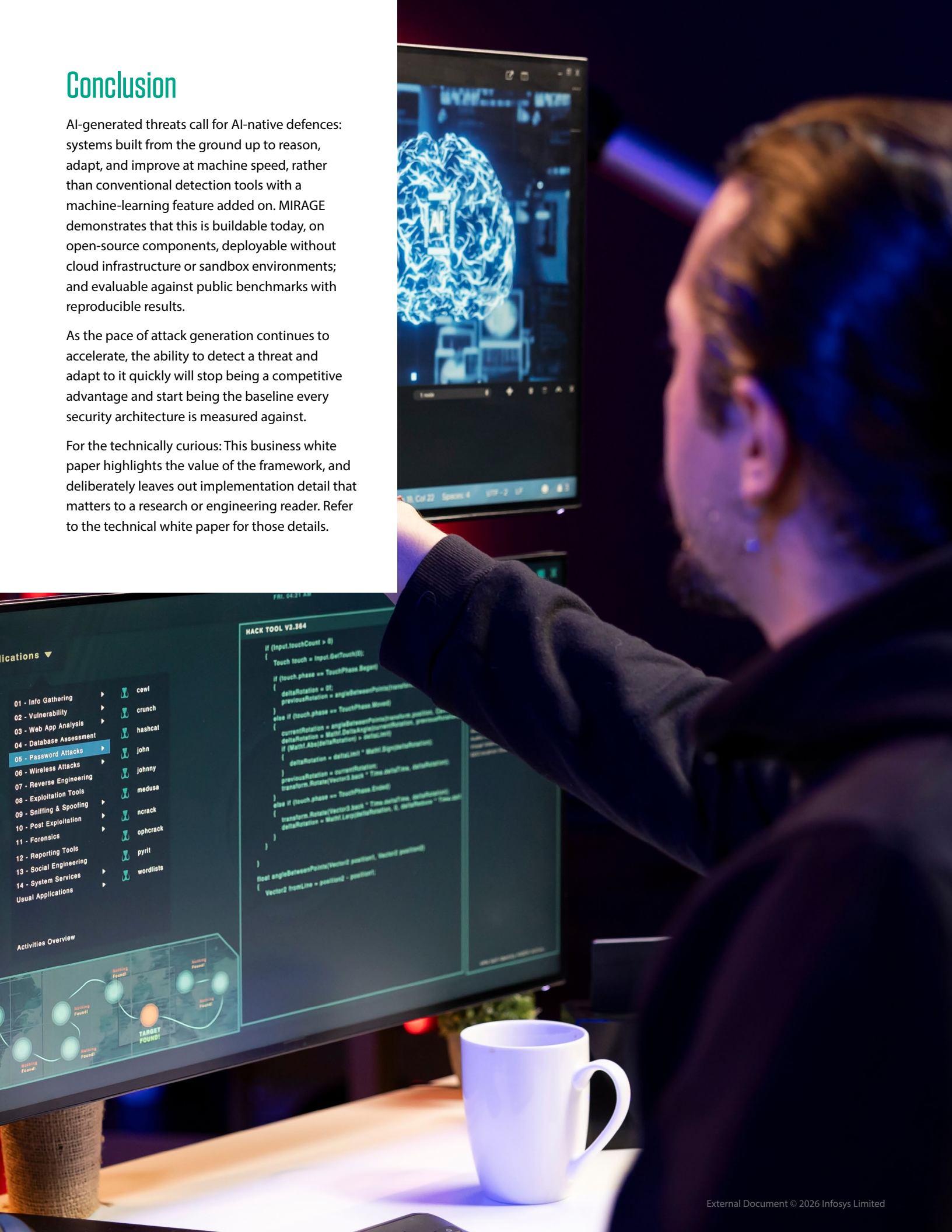
External Document © 2026 Infosys Limited

Conclusion

AI-generated threats call for AI-native defences: systems built from the ground up to reason, adapt, and improve at machine speed, rather than conventional detection tools with a machine-learning feature added on. MIRAGE demonstrates that this is buildable today, on open-source components, deployable without cloud infrastructure or sandbox environments; and evaluable against public benchmarks with reproducible results.

As the pace of attack generation continues to accelerate, the ability to detect a threat and adapt to it quickly will stop being a competitive advantage and start being the baseline every security architecture is measured against.

For the technically curious: This business white paper highlights the value of the framework, and deliberately leaves out implementation detail that matters to a research or engineering reader. Refer to the technical white paper for those details.



External Document © 2026 Infosys Limited

References

1. NIST, "Common Vulnerability Scoring System v3.1," National Vulnerability Database, 2019.
2. E. Raff et al., "Malware Detection by Eating a Whole EXE," AAAI Workshop on Artificial Intelligence for Cyber Security, 2018.
3. H. S. Anderson and P. Roth, "EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models," arXiv:1804.04637, 2018.
4. T. Yang et al., "BODMAS: An Open Dataset for Learning-based Temporal Analysis of PE Malware," IEEE S&P Workshops, 2021.
5. D. Pendlebury et al., "ARES: A System-Oriented Wearable Sensor Dataset for Human Activity Recognition," IEEE T-IFS, 2019. [Concept drift citation to be updated in final version]
6. J. Lu et al., "Learning under Concept Drift: A Review," IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 12, pp. 2346–2363, 2019.
7. A. Ferrag et al., "SecureGPT: LLM-based Autonomous Security Operations," arXiv:2402.00000, 2024. [To be updated with final citation]
8. S. Yao et al., "ReAct: Synergising Reasoning and Acting in Language Models," ICLR, 2023.
9. P. Mell, T. Scarfone, and S. Romanosky, "A Complete Guide to the Common Vulnerability Scoring System," FIRST, 2007.
10. J. Jacobs et al., "Exploit Prediction Scoring System (EPSS)," IEEE S&P, 2021.
11. R. Harang and E. M. Rudd, "SOREL-20M: A Large Scale Benchmark Dataset for Malicious PE Detection," arXiv:2012.07634, 2020.
12. E. J. Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models," ICLR, 2022.
13. C. Guo et al., "On Calibration of Modern Neural Networks," ICML, 2017.
14. Anthropic, "Claude Mythos System Card," 2026. [Online]. Available: https://anthropic.com
15. R. McInnes, J. Healy, and S. Astels, "HDBSCAN: Hierarchical Density Based Clustering," Journal of Open Source Software, 2017.

Author



Malvika Singh is a researcher at the Cyber Applied Research Centre, Infosys, where she focuses on advancing cybersecurity research and developing intelligent solutions to address emerging enterprise security challenges. Her areas of expertise include User and Entity Behavior Analytics (UEBA), Insider Risk Management, Agent Behavior Analysis, Malware Detection, and AI-driven security analytics. She is passionate about leveraging data, machine learning, and behavioral intelligence to strengthen cyber resilience and enable proactive threat detection across complex digital ecosystems.

For more information, contact askus@infosys.com

Infosys
Navigate your next

© 2026 Infosys Limited, Bengaluru, India. All Rights Reserved. Infosys believes the information in this document is accurate as of its publication date; such information is subject to change without notice. Infosys acknowledges the proprietary rights of other companies to the trademarks, product names and such other intellectual property rights mentioned in this document. Except as expressly permitted, neither this documentation nor any part of it may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, printing, photocopying, recording or otherwise, without the prior permission of Infosys Limited and/ or any named intellectual property rights holders under this document.