University of Kent

Kent Academic Repository

ElZemity, Adel, Li, Shujun and Arief, Budi (2026) *Small, Free, and Effective: Orchestrating Open-Weight Small Language Models to Outperform Single LLM for Malware Analysis*. In: Bao, T., Kemerlis, V.P., Meng, W. (eds) Research in Attacks, Intrusions, and Defenses (RAID 2026). Lecture Notes in Computer Science vol 17160 . pp. 61-91. Springer, Cham ISBN 978-3-032-42935-3. E-ISBN 978-3-032-42936-0.

Downloaded from

https://kar.kent.ac.uk/116438/ The University of Kent's Academic Repository KAR

The version of record is available from

https://doi.org/10.1007/978-3-032-42936-0_3

This document version

Author's Accepted Manuscript

DOI for this version

Licence for this version

UNSPECIFIED

Additional information

Versions of research works

Versions of Record

If this version is the version of record, it is the same as the published version available on the publisher's web site. Cite as the published version.

Author Accepted Manuscripts

If this document is identified as the Author Accepted Manuscript it is the version after peer review but before type setting, copy editing or publisher branding. Cite as Surname, Initial. (Year) 'Title of article'. To be published in ***Title of Journal*** , Volume and issue numbers [peer-reviewed accepted version]. Available at: DOI or URL (Accessed: date).

Enquiries

If you have questions about this document contact ResearchSupport@kent.ac.uk. Please include the URL of the record in KAR. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from https://www.kent.ac.uk/guides/kar-the-kent-academic-repository#policies).

Small, Free, and Effective: Orchestrating Open-Weight Small Language Models to Outperform Single LLM for Malware Analysis

Adel ElZemity[0000-0002-5402-7837], Shujun Li[0000-0001-5628-7328], and Budi Arief[0000-0002-1830-1587]

University of Kent, Canterbury, United Kingdom
{ae455, s.j.li, b.arief}@kent.ac.uk

Abstract. Malware analysis demands rapid interpretation of complex detonation reports spanning filesystem, network, and process behaviours. While large language models (LLMs) demonstrate impressive capabilities for technical artifact interpretation, the opacity and escalating API costs of closed-weight frontier models motivate exploration of open-weight alternatives. However, many open-weight models are large, demanding significant compute resources and incurring non-trivial hosting costs that place them beyond reach for resource-constrained deployments. This paper investigates whether orchestrated ensembles of small language models (SLMs) can match or exceed single LLM performance on structured questions about malware detonation reports. We established baselines by testing eleven open-weight SLMs, three cyber security pre-trained models, and six frontier LLMs on Meta's CyberSecEval Malware Analysis benchmark. We then designed and evaluated four orchestration architectures: (i) a multi-agent pipeline that decomposes analysis into structured evidence-collection and reasoning stages, (ii) an adversarial debate framework in which two agents iteratively critique each other's reasoning, (iii) a hierarchical consultation system that pairs a general-purpose SLM with a cyber-specialised expert model, and (iv) a hybrid architecture that combines evidence-grounded pipelines with adversarial debate reasoning. The hybrid system (Qwen3-4B with Foundation-Sec-8B) achieved 35.30% overall accuracy, exceeding the strongest cyber-specialised baseline (22.54%) and the strongest ungrounded frontier baseline (34.77%); when given the same evidence pipeline, grounded Gemini remained the strongest configuration at 38.22%. Case studies on malware from the wild (UNC5142 and Lumma Stealer) illustrated the hybrid system's ability to correct reasoning errors on novel evasion techniques such as EtherHiding and ClickFix. These findings show that evidence-grounded orchestration can substantially improve the performance of collaborative SLMs for supporting interpretation of malware detonation reports.

Keywords: Small language models · Malware analysis · Multi-agent systems · Orchestration · Large language models · Cyber security

1 Introduction

Malware analysis remains a critical bottleneck in cyber security operations. Analysts must rapidly triage suspicious samples, interpret complex detonation reports spanning filesystem modifications, network communications, and process behaviours, and assess threat severity under time pressure [1,4,39]. Traditional static and dynamic analysis pipelines generate rich telemetry—dynamic analysis through sandboxed execution environments and static analysis through disassembly, string extraction, and structural inspection—but extracting actionable intelligence from these multi-faceted reports demands expert knowledge of malware techniques, operating system internals, and attack frameworks such as MITRE ATT&CK [61]. As malware sophistication and volume continue to escalate, the need for automated assistance that can comprehend, reason about, and summarise behavioural evidence has become acute [1,10].

Recent advances in large language models (LLMs) have demonstrated LLMs' capabilities for interpreting technical artifacts and answering domain-specific questions [13]. However, leading closed-weight frontier models remain opaque and costly to deploy via API [25], raising concerns about data privacy, vendor lock-in [56], and the reproducibility of security-critical decisions [32]. At the same time, the open-weight model ecosystem has matured rapidly [30], spanning models from compact, consumer-deployable sizes to large parameter counts that rival closed-weight systems in resource demands. Within this spectrum, increasingly capable small language models (SLMs) have emerged as viable alternatives for specialised tasks without the compute and cost overhead of their larger open-weight counterparts. Following Belcak et al. [6], we use SLM operationally to describe models deployable locally under our fixed single-GPU budget. Our evaluated SLMs contain at most 8B parameters and run at 4-bit quantisation on one RTX 4090; this is a study-specific deployment criterion rather than a definitive SLM–LLM boundary.

While individual SLMs often lag behind frontier LLMs on complex reasoning tasks, orchestration strategies (such as multi-agent systems, debate frameworks, and hierarchical consultation patterns) offer a path to improve SLMs' collective capabilities. Multi-agent architectures have shown promise in decomposing complex tasks into specialised subtasks [68], debate-style interactions can expose reasoning flaws and improve answer quality [17], and hierarchical consultation allows general-purpose models to seek targeted expertise [33]. Open-weight SLMs such as *Mistral*, *Phi*, *Qwen*, and *Llama* variants can be self-hosted, audited, and fine-tuned for domain specificity, while cyber security pre-trained models (ranging from SLM to LLM scale) offer specialised knowledge of malware techniques and defensive concepts [21]. Furthermore, ensembles provide a path to fuse diverse inductive biases while offering defence-in-depth through redundancy [57,45]. Nonetheless, the extent to which orchestrated SLM ensembles can close the performance gap to frontier LLMs on malware analysis tasks remains an open question and it deserves a detailed exploration.

Malware analysts increasingly demand transparent reasoning, reproducible artifacts, and explicit risk controls [58,38,18] when interpreting detonation re-

ports, identifying malicious behaviours, and assessing threat severity [36,48]. These requirements are better served by open-weight models than closed-weight alternatives: weights can be inspected and audited [58,38], local deployment enables greater reproducibility through seed and environment control, and self-hosting allows organisations to implement their own risk control policies over the inference pipeline. However, these advantages only translate into operational value if orchestrated SLM systems can deliver competitive accuracy on real-world malware analysis benchmarks. As such, this paper asks: *Under a fixed single-GPU budget, can orchestrated open-weight SLMs match or exceed single LLMs when answering structured questions about malware detonation reports?* Note that we evaluate report comprehension and behavioural interpretation, rather than static binary reverse engineering or autonomous malware analysis.

This work makes the following key **contributions**:

1. **A systematic evaluation of three orchestration architectures** (agentic, debate, and consult) that instantiate complementary hypotheses for amplifying SLMs' capabilities, evaluated on the CyberSecEval Malware Analysis benchmark across 11 open-weight SLMs and 3 cyber-specialised models.
2. **An empirical characterisation of the interaction between grounding and debate.** Component ablations show that ungrounded debate can introduce drift on retrieval-oriented questions, while evidence grounding stabilises these exchanges and preserves peer-critique gains on harder behavioural-reasoning questions.
3. **An empirical evidence** showing that grounded orchestration benefits both SLMs and LLMs. The open-weight SLM hybrid achieved 35.30% accuracy, compared with 34.77% for the strongest ungrounded frontier baseline, while grounded Gemini achieved 38.22%. This demonstrates that collaborative SLMs can substantially narrow, but do not eliminate, the evidence-matched performance gap.
4. **Qualitative case studies** on malware samples from the wild (UNC5142 EtherHiding campaign and Lumma Stealer with ClickFix), illustrating cases in which the hybrid system is capable of correcting *reasoning errors* on novel evasion techniques that defeat single-model approaches.

The rest of this paper is organised as follows. Section 2 reviews related work in different areas. Section 3 describes our methodology, including the four orchestration architectures and the evaluation benchmark. Section 4 presents experimental results, comparing single-model baselines with orchestrated systems and reporting ablation studies. Section 5 discusses the implications of our findings for operational deployment. Finally, Section 6 concludes our paper and provides several directions for future work.

2 Related Work

Our work builds on four converging research threads: LLMs for cyber security and malware analysis, multi-agent LLM systems, debate frameworks for improving LLM reasoning, and the emerging capabilities of SLMs.

2.1 LLMs for Cyber Security and Malware Analysis

LLMs have been increasingly applied to cyber security tasks, with recent literature documenting their use across vulnerability detection [75], malware analysis and network intrusion detection [69], and threat intelligence [70]. In particular, LLMs have demonstrated promising potential for interpreting malicious artifacts: for instance, Patsakis et al. [49] showed that LLMs achieved 69.56% accuracy in extracting malicious URLs from obfuscated code in real-world campaigns like Emotet, while outperforming symbolic analysis in bypassing common evasion techniques. Al-Karaki et al. [2] presented a comprehensive framework for LLM-based malware detection, identifying key challenges including dataset limitations and the need for domain-specific fine-tuning. The CyberSecEval benchmark suite [65] provides standardised evaluation protocols for assessing LLM capabilities on security tasks, including the malware analysis benchmark we use in this study. While these papers have shown that LLMs can assist with security analysis, they primarily evaluated single models in isolation; our work extends this line by investigating whether orchestration can improve compact-model' performance on answering questions related to detonation-report.

2.2 Multi-Agent LLM Systems

Multi-agent architectures decompose complex tasks across specialised LLM agents that communicate and collaborate. Wu et al. [68] introduced AutoGen, a framework enabling customisable agents with flexible conversation patterns for tasks spanning coding, mathematics, and decision-making. A subsequent work showed that multi-agent orchestration can provide value through deterministic quality and consistency rather than speed alone [62]. Liu et al. [34] proposed dynamic agent networks that optimise team composition based on task requirements, while hierarchical frameworks such as AgentOrchestra [73] use central planning agents that delegate to specialised sub-agents. These systems have shown success in software development, question answering, and enterprise operations. Our agentic and consult systems build on these principles. Related security systems include API-call-based malware frameworks [5,51,64], CVE-Genie for vulnerability reproduction [63], Chimera for insider-threat simulation and log generation [71], and SentinelOne's adversarial-consensus pipeline for tool-assisted binary analysis [60]. In contrast, we evaluate retrieval, consultation, and debate for question answering over existing detonation reports.

2.3 LLM Debate Frameworks

Debate-style orchestration, where multiple LLM agents critique each other's reasoning, has emerged as an effective technique for improving factuality and complex reasoning. Du et al. [17] demonstrated that multi-agent debate improves mathematical and strategic reasoning by exposing flaws through adversarial exchange. Recent extensions include the Mixture-of-Agents framework [66], which organises proposer and aggregator agents in structured layers to achieve

state-of-the-art results using open-source models, and adaptive heterogeneous debate [74], which was shown to have achieved 4–6% accuracy gains over standard debate through dynamic agent weighting. Chen et al. [11] showed that round-table consensus among diverse LLMs can improve reasoning on complex benchmarks. These works established that structured debate improves reasoning quality, but also revealed a trade-off: excessive debate rounds can introduce tangential information that degrades performance on straightforward questions. Our hybrid system addresses this limitation by grounding debate agents in systematically collected evidence, preventing the drift phenomenon while preserving the benefits of peer critique.

2.4 Small Language Models

The capabilities of SLMs – typically defined as models deployable on consumer hardware with acceptable latency [6] – have advanced rapidly. Lu et al. [35] surveyed SLMs in the 100M–5B parameter range, documenting competitive performance on common sense reasoning, mathematics, and domain-specific tasks when compared to much larger models. Recent work [72] demonstrated that well-chosen SLMs can outperform frontier LLMs including GPT-4 variants in specific use cases, particularly when enhanced through fine-tuning, prompt engineering, or ensemble techniques. Recent work on CyberPal 2.0 develops cybersecurity-expert SLMs ranging from 4B to 20B parameters for threat-intelligence and investigation tasks [31]. Our work complements model-development research by evaluating cyber-specialised compact models as orchestration partners for detonation-report question answering.

3 Methodology

Our methodology evaluates whether orchestration improves the performance of open-weight SLMs on structured detonation-report question answering relative to single LLMs. We began by establishing baseline performance: we tested a diverse collection of models (including general-purpose open-weight SLMs, frontier LLMs, and cyber security pre-trained models) as solo agents on the CyberSecEval Malware Analysis benchmark [65]. We selected this benchmark as, to the best of our knowledge, it is the only benchmark designed to automate the evaluation of language models specifically for malware analysis. This benchmark exercises Hybrid Analysis [12] detonation reports[1] through multi-topic, multi-difficulty multiple-choice questions, providing strict accuracy metrics stratified by difficulty tier (Easy, Medium, Hard). The solo-model baselines provided reference points for measuring orchestration gains, and they established which model architectures and parameter scales performed the best on malware analysis tasks when operating independently.

[1] In this work, we take malware analysis detonation reports rather than malware binaries as the input, a standard practice in dynamic malware analysis [48,36].

We tested eleven general-purpose open-weight SLMs spanning 0.6B to 8B parameters: Qwen3-0.6B [54], Llama-3.2-1B [41], Qwen2.5-1.5B-Instruct [52], DeepSeek-R1-Distill-Qwen-1.5B [15], SmolLM2-1.7B [27], Phi-3.5-mini-instruct (3.8B) [43], Gemma-3-4B-IT [23], Qwen3-4B [55], Qwen2.5-Coder-7B-Instruct [53], Ministral-8B [46], and Llama-3.1-8B-Instruct [40]. Their performances were compared against those of three open-weight cyber security pre-trained models spanning SLM to LLM scale (DeepHat-V1-7B [14], Foundation-Sec-8B-Instruct [19], and Llama-Primus-Nemotron-70B [67]), and six frontier LLMs (Gemini 3 Pro Preview [20], Claude Opus 4.5 [3], GPT-5.2 [47], DeepSeek V3.2 [16], Llama 4 Scout, and Llama 4 Maverick [42]).

Open-weight models were sourced from Hugging Face; closed-weight frontier models (Gemini 3 Pro Preview, Claude Opus 4.5, GPT-5.2) were accessed via their official APIs with default sampling parameters. To assess potential contamination, we conducted temporal, reference-corpus overlaps, and answer-probing checks for Foundation-Sec-8B (Appendix C). All SLM inference experiments were conducted on a single NVIDIA RTX 4090 GPU (24 GB VRAM); the hybrid system required approximately 6 GB VRAM with 4-bit quantisation.

Single-pass LLMs might fail on malware analysis tasks through three characteristic failure modes observed during baseline evaluation. First, *context overload*: full Hybrid Analysis JSON reports commonly exceed model context windows, forcing truncation of critical evidence (network telemetry, extracted payloads). Second, *surface-level pattern matching*: models frequently classify samples based on conspicuous keywords (e.g., blockchain terms → mining, Chrome overlays → phishing) rather than tracing causal execution chains. Third, *domain knowledge gaps*: correct interpretation of specific MITRE ATT&CK techniques requires specialised knowledge that general-purpose pre-training does not consistently provide. The four architectures address these failure modes complementarily: the agentic pipeline addresses context overload through structured retrieval; the debate system addresses surface-level reasoning through peer critique; the consult system addresses domain gaps through on-demand expert access; and the hybrid combines all three interventions.

After establishing solo-model baselines, we designed and implemented four distinct orchestration architectures: a specialised multi-agent pipeline (agentic system), an adversarial debate framework (debate system), a hierarchical consultation system (consult system), and a hybrid architecture that combines evidence-grounded pipelines with adversarial debate reasoning (hybrid system). Each architecture embodies a different hypothesis for capability amplification: specialisation through task decomposition, peer critique through adversarial reasoning, expert guidance through hierarchical consultation, and synergistic combination of evidence collection with structured debate. We then evaluated these orchestration systems by running representative SLMs through each architecture on the same malware analysis benchmark. Finally, we compared the orchestrated system performance against the solo-model baselines to quantify the performance gains attributable to orchestration, and to identify which architectural patterns provide the largest improvements across different difficulty tiers and model sizes.

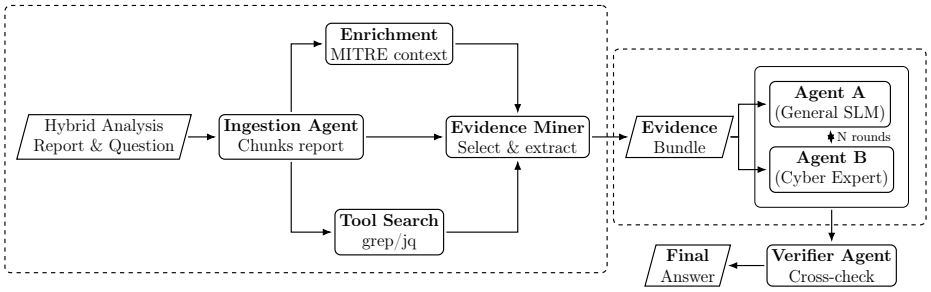


Fig. 1. Hybrid orchestration architecture. *Left (evidence collection phase):* The four-stage agentic pipeline shared with the standalone agentic system (Section 3.1)—ingestion, enrichment, tool-search, and evidence mining—extracts and validates supporting evidence. The standalone agentic system terminates after a single reasoning and verifier step using one model. *Right (debate reasoning phase):* Unique to the hybrid—Agent A (general-purpose SLM) and Agent B (cyber-specialised model) engage in N rounds of evidence-grounded structured debate, followed by a verifier that validates the final answer against the evidence bundle.

Model-internal Mixture-of-Experts [8] routing is outside our scope, which focuses on inference-time orchestration of independently deployable models.

3.1 Agentic System

The first architecture uses a task-oriented multi-agent workflow for interpreting detonation reports. The agentic pipeline is shown in the left panel of Figure 1. The system decomposes the analysis process into six specialised stages orchestrated by a central controller. An *ingestion agent* prepares the workspace by parsing and structuring the relevant malware report: the Hybrid Analysis JSON dossier is divided into semantically coherent sections (process inventory, network telemetry, filesystem modifications, registry changes, and extracted strings) rather than fixed-length windows, preserving field-level context. Each section is chunked at a maximum of 512 tokens with 64-token overlap between adjacent chunks to prevent boundary truncation of multi-field artefacts. (Here "retrieval" refers to selecting relevant chunks from the already-provided report JSON, not fetching from external sources.) An *enrichment agent* augments this data by fetching relevant MITRE ATT&CK technique descriptions [29]. The enrichment trigger policy is heuristic-driven: the agent scans ingested chunks for ATT&CK technique identifiers (e.g., `T1059`, `T1204`) and for a curated vocabulary of 150 technique-indicative keywords (e.g., "persistence", "lateral movement", "credential dumping"), a size determined empirically to balance enrichment coverage against false trigger rate. When a match is detected, the agent queries the MITRE ATT&CK STIX API [61] for the corresponding technique and appends it to the shared evidence state; if no match is found, the enrichment phase is skipped. The enrichment decision is deterministic (keyword/pattern match), not

LLM/SLM-driven, to ensure reproducibility. To locate specific indicators of compromise, a *tool-search agent* generates and executes sandboxed read-only search commands (`grep` for string/pattern matching and `jq` for structured JSON field extraction) against the on-disk report file. The agent receives the benchmark question and the list of ingested chunk summaries as input, generates candidate search queries, executes them in a restricted shell with no network access, and appends raw output to the shared state. An *evidence miner* then extracts supporting snippets from the accumulated report content and assigns each snippet a confidence score $\tau_i \in [0, 1]$, computed as the cosine similarity between the snippet's embedding and the question embedding using `all-MiniLM-L6-v2` [26], a lightweight (22.7M parameter) sentence embedding model designed for semantic similarity and information retrieval, chosen for its computational efficiency and widespread validation in the community. Snippets with $\tau_i > \tau = 0.65$ are retained in the evidence bundle, a threshold determined empirically to balance evidence recall against noise. These retrieved artefacts are synthesised by a *reasoning agent* to formulate an answer, which is finally subjected to a quality gate by a *verifier agent* before being returned. All agents operate on a shared state object that ensures every decision can be traced back to its supporting evidence.

3.2 Debate System

The second architecture pairs two SLM agents (Agent A and Agent B) in a structured adversarial debate. For each question, the agents engage in N rounds: in each round, each agent receives the original prompt plus the full debate history, critiques the opponent's reasoning, and produces a revised rationale and a structured answer. After each round, a controller checks whether the agents agree; if they disagree, it may retrieve and provide relevant MITRE ATT&CK technique descriptions to inform the next round. Once all rounds are complete, the controller produces a final answer by reviewing the full debate transcript. Motivated by security systems that use dedicated verification or sceptical peer review [60,63], the debate tests whether explicit criticism exposes reasoning weaknesses missed in single-pass evaluation.

3.3 Consult System

The third architecture pairs a "tested" general-purpose SLM with a pre-trained cyber security expert model in a hierarchical consultation loop, illustrated in Figure 2. The tested agent owns the task (receives the full benchmark prompt) and may pose at most one sub-question per round; the expert sees only the extracted question and responds with focused domain guidance. An extraction function ψ isolates the sub-question in two stages: it first looks for explicit markers ("Question:", "Query:"), then falls back to the first interrogative sentence longer than 10 words, a minimum length heuristic to exclude trivially short or incomplete questions. Expert guidance accumulates across rounds, allowing the tested agent to iteratively refine its answer without the expert ever controlling the task. This

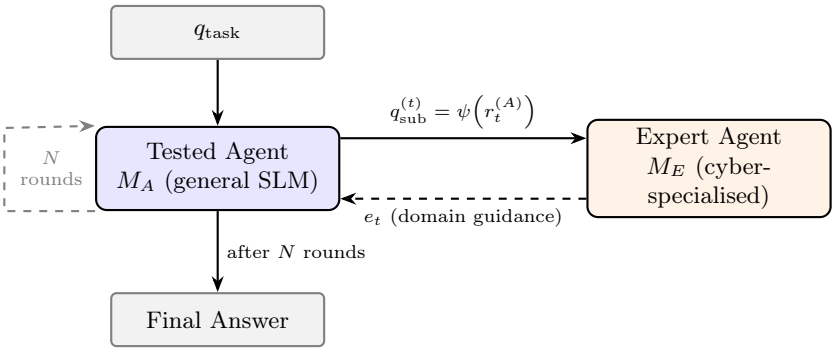


Fig. 2. Consult system architecture. The tested agent (M_A, general-purpose SLM) receives the full benchmark prompt q_{task} and iteratively consults the stateless expert agent (M_E, cyber-specialised), which never sees q_{task}. Each round, ψ extracts a sub-question $q_{\text{sub}}^{(t)}$ from the tested agent's rationale; the expert returns focused domain guidance e_t, which accumulates in H_E across N rounds. The tested agent then produces the final answer.

design probes whether on-demand domain expertise can close the performance gap of compact SLMs lacking deep malware knowledge.

We formalise the consult interaction as a hierarchical loop. Let q_{task} be the full benchmark prompt, M_A the tested agent model, and M_E the expert model. At round t, the tested agent A produces a rationale $r_t^{(A)}$ and potentially a specific query q_{sub} for the expert. An extraction function ψ isolates the explicit question from the tested agent's output. The expert agent E (which does not see q_{task}) provides a domain-specific explanation e_t. The tested agent then updates its state using the accumulated history of expert advice $H_E = \left\{\left(q_{\text{sub}}^{(i)}, e_i\right)\right\}_{i=1}^{t}$:

$$q_{\text{sub}}^{(t)} = \psi\left(r_t^{(A)}\right);\ e_t = M_E\left(q_{\text{sub}}^{(t)}\right);\ r_{t+1}^{(A)} = M_A\left(q_{\text{task}}, H_E\right) \tag{1}$$

This formalisation highlights M_E as a "stateless oracle" relative to the main task, distinguishing this architecture from the state-sharing debate agents.

3.4 Hybrid System

The fourth architecture combines the evidence retrieval capabilities of the agentic system with the adversarial reasoning of the debate system, as illustrated in Figure 1, which shows the full hybrid architecture with the agentic pipeline on the left and the debate reasoning phase on the right. The hybrid system operates in three distinct phases designed to address the complementary weaknesses of its component architectures: evidence collection, adversarial debate reasoning, and final verification.

Phase 1: Evidence Collection. The first four stages of the agentic pipeline execute unchanged: ingestion, enrichment, tool-search, and evidence mining (see Section 3.1 for the full explanation). This phase produces the structured evidence bundle $\mathcal{B}$ formalised below, which serves as a fixed evidence context for the debate reasoning phase.

Formally, let $\mathcal{R}$ denote the raw Hybrid Analysis report (JSON dossier). The *evidence bundle* $\mathcal{B}$ is the union of three extraction outputs: $\phi_{\text{chunk}}(\mathcal{R})$ (ingestion agent's text chunks), $\phi_{\text{enrich}}(\mathcal{R})$ (enrichment agent's external context, e.g., MITRE ATT&CK descriptions), and $\text{Exec}(\phi_{\text{tool}}(\mathcal{R}))$ (results of sandboxed commands generated by the tool-search agent). The evidence miner applies a filtering function F_τ based on a confidence threshold $\tau = 0.65$ (determined empirically; see Section 3.1) to produce the final validated evidence set E_{final}, which serves as the fixed evidence context for the debate reasoning phase:

$$\mathcal{B} = \phi_{\text{chunk}}(\mathcal{R}) \cup \phi_{\text{enrich}}(\mathcal{R}) \cup \text{Exec}(\phi_{\text{tool}}(\mathcal{R})) \tag{2}$$

$$E_{\text{final}} = \{e \in \mathcal{B} \mid \text{Confidence}(e) > \tau\} \tag{3}$$

Phase 2: Debate Reasoning. Rather than passing evidence to a single reasoning agent, the hybrid system instantiates two debate agents that receive the collected evidence alongside the original question. For instance, Agent A is a general-purpose SLM (Qwen3-4B), selected based on its strong baseline performance across all orchestration systems; Agent B is a cyber-specialised model (Foundation-Sec-8B), selected to provide complementary domain expertise while maintaining capacity balance (within 2× parameter ratio). The agents engage in N rounds of structured debate following the protocol described in Section 3.2, but with a critical modification: agents are explicitly instructed to cite collected evidence when defending their positions and to challenge claims that lack evidential support.

We model the debate as a Markov process over t rounds. In round t, Agent A's response $r_t^{(A)}$ is conditional on the original question q, the evidence bundle E_{final}, the debate history H_{t-1}, and the opponent's previous argument $r_{t-1}^{(B)}$:

$$r_t^{(A)} = M_A\left(q, E_{\text{final}}, H_{t-1}, r_{t-1}^{(B)}\right) \tag{4}$$

Unlike standard debates, the hybrid system enforces a *grounding constraint* via the verifier: a response r is valid if and only if every claim $c \in r$ maps to a supporting snippet in E_{final} with cosine similarity $S(c, e)$ (computed using the same embedding model as τ) exceeding threshold $\lambda = 0.55$, set lower than τ since claims in a debate response are often paraphrases or inferences from the evidence rather than direct matches, requiring a more relaxed similarity criterion:

$$\text{Valid}(r) \iff \forall c \in r,\ \exists e \in E_{\text{final}} \text{ s.t. } S(c, e) \geq \lambda \tag{5}$$

This constraint prevents the "drift" phenomenon observed in pure debates, where agents introduce tangential information that degrades easy-questions' accuracy.

Phase 3: Verification. After the debate concludes, the verifier agent validates the debate conclusion against the evidence bundle, checking that the selected answer has supporting evidence with confidence above λ and that the reasoning chain is internally consistent. If verification fails, the system falls back to the answer most directly supported by the evidence bundle, bypassing the debate conclusion to ensure that the final output is always grounded in collected evidence.

4 Results

This section presents the experimental findings of our study. Throughout, the task context is that of a Security Operations Centre (SOC) analyst querying a malware detonation report — the primary real-world setting in which such reports are interpreted under time pressure: given a Hybrid Analysis JSON dossier, can the system correctly answer structured questions about the sample's persistence mechanisms, network behaviour, and evasion techniques? We begin with single-model baselines across various parameter scales to establish single-model reference points. We then detail the performance of the four orchestrated architectures (agentic, debate, consult, and hybrid) to quantify the capability gains achieved through different ensemble strategies. We conclude with qualitative case studies of real-world malware samples from the wild, demonstrating the hybrid system's ability to identify and reason through novel evasion techniques.

4.1 Benchmark and Evaluation Protocol

We ground our experiments in the CyberSecEval Malware Analysis benchmark (CyberSOCEval test suite) [13], which pairs Hybrid Analysis detonation reports with multi-topic, multi-difficulty multiple-choice questions spanning evidence retrieval, behavioural interpretation, risk scoring, and system-interaction audits. The multi-label format (up to nine answer options per question) penalises both omissions and incorrect option selections, providing exact-match accuracy and Jaccard partial-credit scores stratified by topic, difficulty tier, and malware family. We reserve *hallucination* for rationales containing claims unsupported by the supplied report or enrichment evidence. Ground-truth options are isolated from model inputs, so systems must locate and reconcile the relevant report fragments rather than memorise expected outputs.

A Benchmark Example. A typical Medium-difficulty question asks which persistence mechanisms a sample uses (multi-select from 9 options); the correct answer is embedded within thousands of lines of process telemetry. A single-pass LLM must locate and cross-reference the relevant report fragments without retrieval; the agentic system uses `jq` and `grep` to retrieve them directly.

Let $D = \{\text{Easy}, \text{Medium}, \text{Hard}\}$ denote the difficulty tiers, N_d the number of questions in Tier d, and Acc_d the model's accuracy on that tier; $N_{\text{total}} = 609$ ($N_{\text{Easy}} = 451$, $N_{\text{Med}} = 136$, $N_{\text{Hard}} = 22$). The *overall weighted accuracy* is

Table 1. Accuracy percentages for each model on the Malware Analysis benchmark, stratified by difficulty level.

Model	Params	Easy ($n = 451$)	Medium ($n = 136$)	Hard ($n = 22$)	Overall ($n = 609$)
LLMs					
Llama 4 Scout	109B	25.50%	14.00%	0.00%	22.01%
Llama 4 Maverick	400B	31.00%	20.50%	13.64%	28.03%
DeepSeek V3.2	685B	33.50%	22.00%	13.64%	30.21%
Claude Opus 4.5	—	36.25%	24.75%	21.59%	33.15%
Gemini 3 Pro Preview	—	**38.00%**	**26.00%**	**22.73%**	**34.77%**
GPT-5.2	—	34.50%	23.50%	20.45%	31.54%
Cyber security language models					
DeepHat-V1-7B	7B	16.41%	11.03%	4.55%	14.78%
Foundation-Sec-8B	8B	**22.65%**	**13.45%**	**4.55%**	**19.96%**
Llama-Primus-Nemotron-70B	70B	24.78%	18.00%	4.55%	22.54%
SLMs					
Qwen3-0.6B	0.6B	12.94%	6.61%	0.00%	11.05%
Llama-3.2-1B	1B	10.17%	5.88%	0.00%	8.87%
Qwen2.5-1.5B-Instruct	1.5B	11.72%	7.35%	0.00%	10.34%
DeepSeek-R1-Distill-Qwen-1.5B	1.5B	12.42%	8.09%	4.55%	11.17%
SmolLM2-1.7B	1.7B	10.86%	6.62%	0.00%	9.52%
Phi-3.5-mini-instruct	3.5B	15.74%	10.29%	4.55%	14.12%
Gemma-3-4B-IT	4B	16.62%	11.03%	4.55%	14.94%
Qwen3-4B	4B	**18.40%**	**12.50%**	4.55%	**16.58%**
Qwen2.5-Coder-7B-Instruct	7B	12.70%	9.19%	4.55%	11.66%
Ministral-8B	8B	11.75%	8.82%	4.55%	10.84%
Llama-3.1-8B-Instruct	8B	13.24%	9.56%	4.55%	12.15%

calculated as follows:

$$\mathrm{Acc}_{\mathrm{overall}} = \sum_{d \in D} \frac{N_d}{N_{\mathrm{total}}} \cdot \mathrm{Acc}_d = \frac{451 \cdot \mathrm{Acc}_{\mathrm{Easy}} + 136 \cdot \mathrm{Acc}_{\mathrm{Med}} + 22 \cdot \mathrm{Acc}_{\mathrm{Hard}}}{609} \quad (6)$$

Because Easy questions constitute 74% of the dataset, performance on this tier dominates the overall score, i.e., a model's ability to handle straightforward evidence-retrieval queries has a greater impact on its aggregate ranking than proficiency on the rare, complex Hard questions. This weighting must be kept in mind when interpreting overall accuracy figures: a system that excels on Hard questions but degrades on Easy ones may rank lower overall, even if it demonstrates superior reasoning capability on the most challenging cases. We therefore report per-tier accuracy alongside overall scores throughout this section.

Model Selection for Orchestration Experiments. Qwen3-4B is selected as the general-purpose agent based on its leading performance across all solo and orchestrated configurations; Foundation-Sec-8B provides cyber-specialised expertise within the optimal capacity balance. The three selection criteria (capacity balance, complementary expertise, strong baseline) are discussed in Section 5.

4.2 Single-Model Baselines

Table 1 contrasts representative LLM and SLM baselines across benchmark difficulty tiers, establishing the empirical gap our orchestration aims to close.

Table 2. Comparison of orchestrated systems against single-model baselines on the Malware Analysis benchmark. Orchestrated systems are tested with four representative SLMs: Qwen3-0.6B, Phi-3.5-mini-instruct (3.5B), Qwen3-4B, and Ministral-8B. *Agentic* system uses a single SLM with sandboxed command-line tools; *Debate* pairs each SLM with Foundation-Sec-8B (seven rounds); *Consult* uses each SLM as tested agent consulting Foundation-Sec-8B as cyber expert (seven rounds); *Hybrid* combines agentic evidence collection with debate reasoning (seven rounds).

System Model	Easy ($n = 451$)	Medium ($n = 136$)	Hard ($n = 22$)	Overall ($n = 609$)
Single-model baselines				
Best open-weight SLM (Qwen3-4B)	18.40%	12.50%	4.55%	16.58%
Best single LLM (Gemini 3 Pro Preview)	**38.00%**	**26.00%**	**22.73%**	**34.77%**
Agentic (with tools)				
Qwen3-0.6B	22.62%	17.65%	9.09%	21.02%
Phi-3.5-mini-instruct	24.61%	19.85%	13.64%	23.15%
Qwen3-4B	**26.50%**	**21.30%**	**20.00%**	**25.11%**
Ministral-8B	25.71%	20.59%	18.18%	24.30%
Debate (7 rounds, paired with Foundation-Sec-8B)				
Qwen3-0.6B	20.00%	14.80%	13.64%	18.60%
Phi-3.5-mini-instruct	22.40%	17.70%	18.18%	21.20%
Qwen3-4B	**25.50%**	**19.80%**	**22.73%**	**24.13%**
Ministral-8B	25.00%	19.10%	**22.73%**	23.60%
Consult (7 rounds, paired with Foundation-Sec-8B)				
Qwen3-0.6B	21.51%	16.18%	**9.09%**	19.87%
Phi-3.5-mini-instruct	23.06%	18.38%	**9.09%**	21.51%
Qwen3-4B	**24.50%**	**19.10%**	**9.09%**	**22.74%**
Ministral-8B	23.95%	18.38%	**9.09%**	22.17%
Hybrid (evidence-informed debate, 7 rounds)				
Qwen3-0.6B + Foundation-Sec-8B	28.82%	19.85%	18.18%	26.44%
Phi-3.5-mini-instruct + Foundation-Sec-8B	34.59%	24.26%	22.73%	31.86%
Qwen3-4B + Foundation-Sec-8B	**38.14%**	**27.21%**	**27.27%**	**35.30%**
Ministral-8B + Foundation-Sec-8B	36.59%	25.74%	**27.27%**	33.83%

The baseline profiling revealed that *parameter count alone does not predict malware-analysis capability*. For instance, Qwen3-4B achieved 16.58% overall accuracy, outperforming all tested 7–8B models, while sub-2B models cluster tightly in the 8.87–11.17% range with minimal performance differences.

Based on this profiling, we selected four representative SLMs for detailed orchestration experiments: Qwen3-0.6B, Phi-3.5-mini-instruct (3.5B), Qwen3-4B, and Ministral-8B. These models span four distinct size classes (sub-1B, mid-range 3.5B, mid-range 4B, and 8B), representing diverse architectural families (Qwen, Phi, and Mistral variants), and demonstrating strong baseline performance within their respective categories: Qwen3-0.6B achieved the highest overall accuracy (11.05%) among sub-1B models, Phi-3.5-mini-instruct delivered competitive mid-tier performance (14.12%), Qwen3-4B achieved the best overall performance among all open-weight SLMs (16.58%), and Ministral-8B provided an 8B reference point (10.84%). This selection enabled us to assess whether the benefits of orchestration generalise across model scales and whether architectural diversity influences ensemble effectiveness.

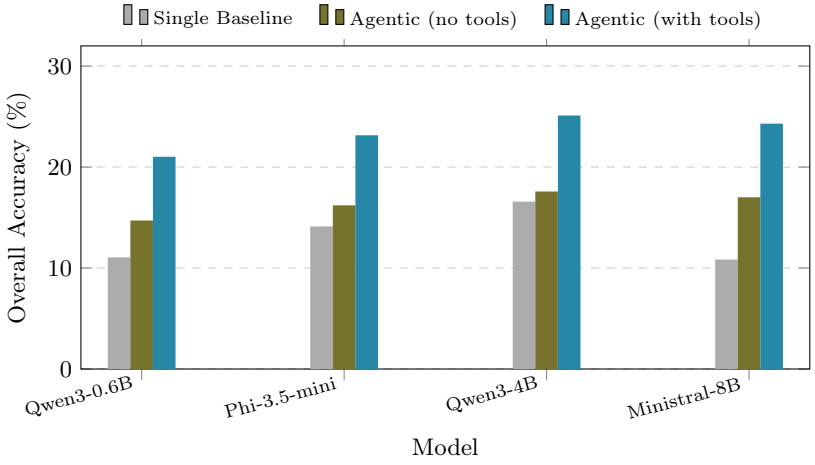


Fig. 3. The ablation analysis of the agentic system showing the impact of command-line tools across four representative SLMs. Tool access provides the largest performance boost for all models. Qwen3-4B with tools achieves the best overall performance among agentic configurations (25.11%), surpassing all open-weight SLM solo baselines.

4.3 Orchestrated Systems Performance

Table 2 presents the performance of all three orchestrated systems against their single-model baselines, demonstrating the gains achieved through orchestration.

Agentic System Results The agentic multi-stage pipeline shows that tool-augmented workflows can materially upgrade SLM capabilities. Across all four representative models (Qwen3-0.6B, Phi-3.5-mini-instruct, Qwen3-4B, Ministral-8B), the agentic system consistently outperformed their single-model baselines, with ablation studies (Figure 3) indicating that access to carefully sandboxed command-line tools provided the largest incremental boost to overall performance for each model. Notably, Qwen3-4B with tools achieved 25.11% overall accuracy, surpassing all open-weight SLM solo baselines and exceeding the weakest LLM baseline (Llama 4 Scout at 22.01%), though it fell below the stronger frontier models, demonstrating that tool-augmented evidence retrieval alone is insufficient to close the gap to the strongest LLM baselines and that additional reasoning mechanisms — as provided by the hybrid system — are necessary to exceed the best single LLM baseline (Gemini 3 Pro Preview at 34.77%).

Debate System Results When we applied the debate-style orchestration, pairing each of the four representative SLMs with the cyber-specialised Foundation-Sec-8B model, we observed a complementary set of effects that held consistently across all tested configurations (Figure 4). Increasing the number of debate rounds consistently improved performance on the *hard* questions, but degraded

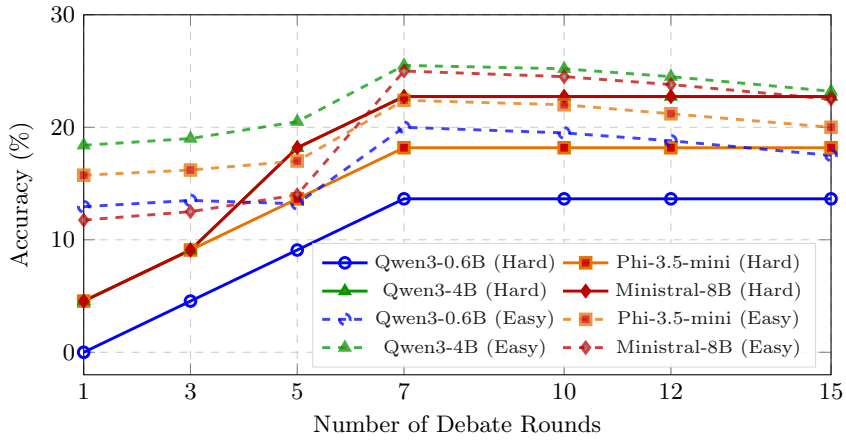


Fig. 4. The debate system's performance as a function of debate rounds (1–15 rounds), where each SLM debates with Foundation-Sec-8B. Results show consistent improvement on hard questions (solid lines) that plateaus after 7–10 rounds, while easy questions (dashed lines) exhibit clear degradation with increased rounds, declining from peak performance at rounds 7–10 to lower accuracy by round 15. Qwen3-4B achieves the highest performance across debate rounds.

accuracy on the easiest items; inspection of the logs showed that introducing irrelevant or tangential evidence into the debate sometimes pulled initially correct answers towards incorrect alternatives. Across all tested pairings, performances on hard questions saturated after roughly 7–10 rounds, while accuracy on easy questions continued to decline with additional rounds beyond this point. The debate system achieved overall scores that exceeded all open-weight SLM solo baselines when operated at the optimal round count, though they remain below the strongest frontier LLM baselines. Among the representative models, Qwen3-4B paired with Foundation-Sec-8B achieved the highest debate performance (24.13% overall), demonstrating that mid-sized models with strong baseline capabilities can engage effectively in multi-round critique.

The full all-pairs debate-partner matrix and extended interpretation are provided in Appendix E (Table 7); in summary, results remain symmetric by role assignment, favour complementary general+cyber pairings, and suggest an association between large parameter imbalance and lower performance.

Consult System Results Consulting a cyber-specialised expert (Foundation-Sec-8B) consistently outperformed consulting a general-purpose LLM across all four tested SLMs, but scaling the expert to 70B yielded minimal additional gain (see Figure 5). Qwen3-4B achieved the highest consult performance (22.74% overall), the lowest among all four orchestration architectures (see Table 2).

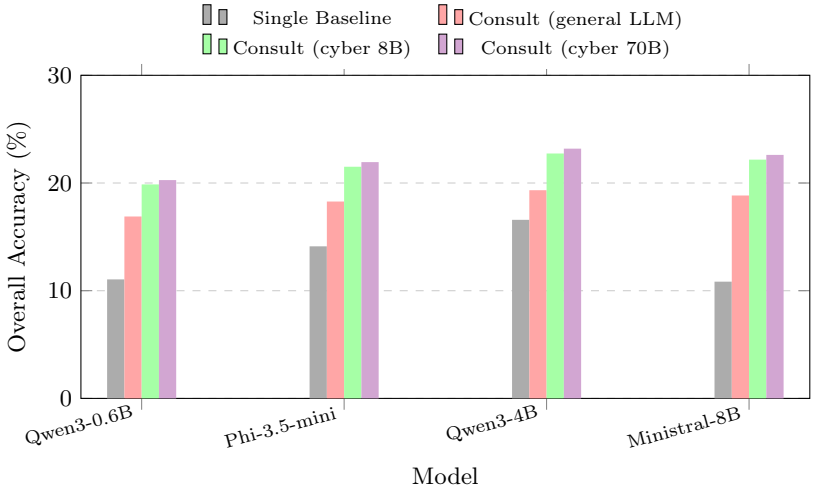


Fig. 5. The consult system's performance comparison where each SLM consults Foundation-Sec-8B (cyber 8B) versus a general LLM or a larger 70B cyber expert. Cyber-specialised experts consistently outperform general LLMs, but parameter scaling from 8B to 70B yields minimal additional gains. Qwen3-4B achieves the highest consult performance (22.74% overall with Foundation-Sec-8B).

Hybrid System Results The hybrid system achieved the strongest overall performance among the evaluated open-weight configurations (see Table 2). When Qwen3-4B and Foundation-Sec-8B were paired, they achieved 35.30% overall accuracy, surpassing both the strongest ungrounded single-LLM baseline (Gemini 3 Pro Preview at 34.77%) and the best individual orchestration systems (agentic at 25.11%, debate at 24.13%, consult at 22.74%). Performance gains were observed across all difficulty tiers:

- **Easy**: 38.14% (vs. 26.50% agentic, 25.50% debate, 38.00% best ungrounded LLM)
- **Medium**: 27.21% (vs. 21.30% agentic, 19.80% debate, 26.00% best ungrounded LLM)
- **Hard**: 27.27% (vs. 20.00% agentic, 22.73% debate, 22.73% best ungrounded LLM)

The hybrid system addressed the easy-question degradation observed in pure debate (Section 3.2). By grounding debate agents in systematically collected evidence, the system prevented the introduction of tangential information that previously pulled correct answers toward incorrect alternatives. Simultaneously, the debate reasoning phase preserved the reasoning improvements that peer critique provided on hard questions. Ablation studies revealed that removing either component degrades performance: omitting evidence collection reduced easy-tier accuracy by 12 percentage points (reverting to pure debate behaviour),

Table 3. Component ablation for the hybrid system (Qwen3-4B + Foundation-Sec-8B). Each row removes one component from the full hybrid. "−evidence collection" replaces Phase 1 with direct report provision (pure debate). "−grounding constraint" removes the verifier's evidence-citation check. "−debate" replaces Phase 2 with single-pass reasoning (pure agentic). "−verifier" removes the final validation step.

Configuration	Easy	Medium	Hard	Overall
Full hybrid (Qwen3-4B + Sec-8B)	38.14%	27.21%	27.27%	35.30%
− evidence collection (pure debate)	25.50%	19.80%	22.73%	24.13%
− grounding constraint	36.21%	25.81%	26.14%	33.79%
− debate (pure agentic)	26.50%	21.30%	20.00%	25.11%
− verifier	37.82%	26.93%	26.52%	35.04%

while replacing debate with single-pass reasoning reduced hard-tier accuracy by 7 percentage points (reverting to pure agentic behaviour).

Substituting Ministral-8B for Qwen3-4B yielded 33.83% accuracy, indicating that the hybrid gain was not limited to Qwen3-4B among the tested pairings. Pairing Qwen3-0.6B with Foundation-Sec-8B yielded 26.44%; because parameter ratio covaries with solo accuracy, architecture, and specialisation, this observation motivates rather than validates the capacity-balance heuristic.

4.4 Ablation Study

Table 3 decomposes the hybrid system's performance by removing one component at a time. The "−debate" row (pure agentic) reused values from Table 2. Removing evidence collection (reverting to pure debate) reduced Easy-tier accuracy from 38.14% to 25.50% (−12.64 percentage points), indicating that evidence grounding influences easy-question performance and that pure debate can introduce tangential drift on straightforward retrieval tasks. Removing debate (reverting to pure agentic) reduced Hard-tier accuracy from 27.27% to 20.00% (−7.27 percentage points), suggesting that peer critique contributes to complex reasoning improvements. Removing the grounding constraint modestly degraded the performance (38.14% → 36.21% Easy; 35.30% → 33.79% overall), suggesting the constraint provides measurable benefit by preventing un-cited claims from corrupting easy-question answers. Removing the verifier had a minimal impact on accuracy (35.30% → 35.04%), in agreement with its role as a consistency gate rather than a primary performance driver.

4.5 Grounded LLM Comparison

To address the fairness concern that SLMs receive tool-augmented grounding while frontier LLMs do not, Table 4 presents frontier LLMs run through the full hybrid orchestration pipeline with identical evidence collection. This experiment directly tested whether the orchestration architecture can provide benefits that extend to frontier models. Grounding frontier LLMs through the hybrid pipeline yielded gains: Gemini 3 Pro Preview improved from 34.77% to

Table 4. Frontier LLMs under hybrid orchestration vs. ungrounded single-pass baseline. "Hybrid (LLM)" runs the frontier model as Agent A in place of the SLM, with full evidence collection Phase 1 and Foundation-Sec-8B as Agent B. This tests whether orchestration benefits are model-size-agnostic.

System	Model	Easy	Hard	Overall
Ungrounded single-pass (baseline)				
	Gemini 3 Pro Preview	38.00%	22.73%	34.77%
	Claude Opus 4.5	36.25%	21.59%	33.15%
Hybrid orchestration (grounded)				
	Gemini 3 Pro Preview	41.24%	29.55%	38.22%
	Claude Opus 4.5	39.29%	28.68%	36.85%
SLM hybrid (for reference)				
	Qwen3-4B + Sec-8B	38.14%	27.27%	35.30%

38.22% (+3.45 percentage points) and Claude Opus 4.5 from 33.15% to 36.85% (+3.70 percentage points), showing gains for both evaluated frontier models. Grounded Gemini (38.22%) outperformed the SLM hybrid (35.30%), indicating that frontier models extract value from structured evidence when their reasoning capacity is greater. The SLM hybrid reached 35.30%, below grounded Claude Opus 4.5 at 36.85%, while incurring no API charges ($0.00 vs. $96.22 per benchmark run). These results reframe the contribution: the hybrid orchestration architecture is a general-purpose evidence-grounded reasoning framework that benefits all model sizes, with open-weight SLMs offering a locally deployable path that narrows the gap to proprietary alternatives on structured detonation-report questions.

4.6 Case Studies: Qualitative Analysis of Samples from the Wild

For a preliminary assessment beyond multiple-choice benchmarks, we evaluated the hybrid system on 12 malware samples from public threat intelligence feeds (January 2026) exhibiting novel evasion techniques not present in CyberSecEval [7]. The hybrid system correctly classified 9 of 12 samples (75.0%) versus 5 of 12 (41.7%) for the single-model baseline (Gemini 3 Pro Preview). In both representative cases (UNC5142 EtherHiding and Lumma Stealer ClickFix), Phase 1 extracted obscure artifacts that the single model missed—blockchain payload fields and clipboard event handlers respectively—while Phase 2 debate corrected surface-level reasoning errors. Full evaluation methodology, per-sample scoring, and case narratives are provided in Appendix D.[2]

5 Further Discussions

Although evidence-grounded orchestration improved performance, the best exact-match accuracy of 35.30% remains insufficient for autonomous malware-analysis

[2] Sample SHA256 hashes, Hybrid Analysis report identifiers, full system traces, and raw LLM outputs are available in the anonymous repository accompanying this paper (`https://github.com/Adelsamir01/slms_mal`).

Table 5. Latency and API-charge comparison. Latency is the mean wall-clock time across 609 questions; cost is the estimated API charge for the full benchmark run and excludes local hardware, energy, hosting, and support.

System	Latency (s/question)	Est. API Cost (USD/609 Qs)
Single-model baselines		
Gemini 3 Pro Preview (API)	3.2	$7.54
Claude Opus 4.5 (API)	4.8	$96.22
GPT-5.2 (API)	3.9	$57.86
Qwen3-4B (local)	1.8	$0.00
Orchestrated SLM systems (local, RTX 4090)		
Agentic (Qwen3-4B)	38.4	$0.00
Debate (Qwen3-4B + Sec-8B)	67.2	$0.00
Consult (Qwen3-4B + Sec-8B)	53.8	$0.00
Hybrid (Qwen3-4B + Sec-8B)	105.6	$0.00

decisions. Our results therefore support an analyst-assistance setting in which retrieved evidence and model outputs require human verification; operational effectiveness and alert fatigue remain subjects for future evaluation.

The results reveal complementary strengths across orchestration approaches for malware analysis tasks. Tool-augmented agentic systems excel on straightforward evidence retrieval from detonation reports (Easy questions); debate systems dramatically improve complex behavioural interpretation and multi-step reasoning (hard-tier accuracy from 4.55% baseline to 22.73%); while consult systems provide consistent moderate gains by injecting domain-specific malware expertise. The hybrid system synthesises these complementary strengths: evidence collection grounds the debate in retrieved artifacts, preventing the tangential drift that degrades easy-question accuracy in pure debate, while the debate phase preserves the reasoning improvements that peer critique provides on hard questions. This synergy yields the first orchestrated SLM configuration to exceed all ungrounded frontier LLM baselines across all difficulty tiers simultaneously.

The hybrid result is consistent with three empirically observed partner-selection criteria: capacity balance (within an approximately 4× parameter ratio), complementary general and cyber expertise, and strong solo performance. However, parameter ratio covaries with baseline accuracy, architecture, and specialisation in our experiments; the 4× value should therefore be treated as a benchmark-specific heuristic rather than a validated threshold.

5.1 Cost and Latency Analysis

Table 5 compares per-question latency and API charges. Local inference used one NVIDIA RTX 4090 at 4-bit quantisation and incurred no API charges. The hybrid achieved 35.30% accuracy at 105.6 s per question, compared with Gemini at 34.77%, 3.2 s, and $7.54 per run, and Claude at 33.15%, 4.8 s, and $96.22 per run. This is not a total-cost-of-ownership analysis: the local figures exclude electricity, hardware acquisition and amortisation, hosting, and support.

Key considerations for deployment in operational malware analysis workflows include governance over orchestration policies, monitoring for correlated

failure modes when analysing polymorphic threats, and ensuring that human analysts remain in the decision loop for high-confidence classifications. Open-weight ensembles afford adaptability and jurisdictional control but require disciplined Machine Learning Operations (MLOps) practices to manage model drift and dependency chains. The hybrid architecture's two-phase design also provides natural checkpoints for human reviews: analysts can inspect the evidence bundle before debate and intervene if critical artifacts are missing.

5.2 Limitations

This work evaluates *comprehension* of Hybrid Analysis detonation reports, not low-level binary analysis; findings should be generalised accordingly. The multiple-choice format differs from open-ended triage, while our 12-sample, single-evaluator case study provides only preliminary qualitative evidence. Because this is a question-answering benchmark rather than a malware-detection task, exact-match accuracy should not be interpreted as a detection rate; operational precision and recall are outside the benchmark's design. Off-the-shelf coding agents were not compared because their performance depends on the supplied tools and environment; defining an equivalent detonation-report workflow remains future work. Sandbox evasion techniques can produce incomplete reports where questions are unsolvable regardless of model capability; performance gains may also vary as the open-weight ecosystem evolves. Because retrieval uses keyword triggers and embedding similarity, adversarial tricks (such as decoys, keyword stuffing, or event flooding) could promote misleading snippets or suppress relevant evidence; we did not evaluate such perturbations.

6 Conclusion and Future Work

This paper evaluated whether orchestration improves compact open-weight models when answering structured questions about malware detonation reports. The hybrid achieved 35.30% accuracy, exceeding the strongest ungrounded frontier baseline at 34.77% but remaining below grounded Gemini at 38.22%. These results indicate that evidence collection and peer critique narrow the performance gap under local deployment, although the observed accuracy remains insufficient for autonomous decisions and requires analyst verification. Future work will investigate transfer to other security domains, dynamic difficulty-based routing, and human-in-the-loop studies of analyst trust and effectiveness.

Acknowledgments. This work was partly supported by the UK EPSRC project grant EP/X036707/1 on Countering HArms caused by Ransomware in the Internet Of Things (CHARIOT).

A Ethical Considerations

This work investigates orchestrated ensembles of open-weight small language models for analyst-assisted interpretation of malware detonation reports. We

address the ethical dimensions of this research following the principles outlined in the Menlo Report and standard ACM/IEEE research ethics guidelines.

Benefits and Potential Harms. The potential benefit is a transparent, auditable, and locally deployable tool that assists analysts with detonation-report interpretation. The open-weight pipeline supports organisational control and data-sovereignty requirements, but its outputs require human verification and should not be used autonomously. The potential harm we considered is dual-use: the orchestration architectures we describe could theoretically be adapted by threat actors to improve malware generation or evasion techniques. However, we assess this risk as low because (1) the techniques we present are defensive in nature, focused on interpreting existing malware behaviour rather than generating novel attacks; (2) the orchestration patterns (multi-agent pipelines, debate, consultation) are already documented in the broader LLM literature; and (3) the primary barrier to malware development is not reasoning capability but rather access to delivery infrastructure and operational security knowledge, which our work does not address.

Data Sourcing and Privacy. All malware samples analysed in this work were obtained from publicly accessible sources. The CyberSecEval Malware Analysis benchmark uses Hybrid Analysis detonation reports that are publicly available. The samples from the wild evaluated in Section 4.6 were collected from public threat intelligence feeds and represent malware campaigns that have been extensively documented in prior security research (UNC5142, Lumma Stealer). No private victim data was accessed or analysed. The Hybrid Analysis reports we processed contain behavioural telemetry from sandboxed detonations, not data from real victim systems. We did not interact with any live command-and-control infrastructure or active malware campaigns.

Responsible Disclosure. Our research did not discover new vulnerabilities in software or systems. The malware techniques discussed (EtherHiding, ClickFix) were already publicly documented by security vendors prior to our analysis. We did not develop or release any offensive capabilities, malware samples, or exploitation tools.

Experimental Safety. All experiments were conducted in isolated environments. SLM inference was performed on local hardware without network access to external systems beyond model weight downloads. The tool-augmented agentic system executes only sandboxed read-only commands (grep, jq) on static report files; no commands were executed on live systems or with elevated privileges.

Deployment Considerations. We emphasise that analyst-assistance systems such as the hybrid architecture, should augment rather than replace human analyst judgement. Section 5 of our paper explicitly recommends that human analysts remain in the decision loop for high-confidence classifications and that organisations implement governance policies for orchestration system deployment. The two-phase architecture provides natural checkpoints for human review.

B Open Science

All code and configuration used in this work are released in an anonymous open repository at `https://github.com/Adelsamir01/slms_mal`. The repository contains the implementations of the agentic, debate, consult, and hybrid orchestration systems, together with experiment harnesses for the CyberSecEval Malware Analysis benchmark and scripts to regenerate all reported tables and figures. Readers can clone or download the repository from this URL and follow the instructions in the top-level documentation to set up the environment, run the evaluation pipeline, and verify our results.

C Data Contamination Audit

To address the critical issue of test set leakage in Large Language Model evaluation, we performed a three-stage decontamination audit on our primary expert agent, Foundation-Sec-8B-Instruct.

C.1 Temporal Sanity Check

The Foundation-Sec-8B-Instruct model reports a strict knowledge cutoff of April 10, 2025 [19].

- **Benchmark integrity:** The specific Hybrid Analysis detonation reports used in the CyberSecEval test split were generated dynamically for the evaluation suite and are not present in the public Common Crawl.
- **Validity of samples taken from the wild:** The malware samples analysed in Section 4.6 (e.g., Lumma Stealer variants) were collected from active campaigns in late 2025, months after the model's training window closed. This temporal gap guarantees that the expert agent could not have memorised these specific threat artifacts during pre-training.

C.2 *n*-Gram Overlap Analysis

Following Carlini et al. [9] and Golchin et al. [22], we compared the benchmark question stems with the public threat-intelligence reference corpus assembled for this audit. Because the model's complete training corpus is not disclosed, this analysis cannot exclude overlap with undisclosed training data. We define the n-gram overlap ratio as:

$$\text{Overlap}_n(T, C) = \frac{|\{g \in \text{ngrams}_n(T) : g \in C\}|}{|\text{ngrams}_n(T)|}, \tag{7}$$

where T is the set of benchmark question stems and C is the audited reference corpus. We use $n = 13$ following prior work, as 13-grams are long enough to detect meaningful memorisation while avoiding false positives from common phrases.

We found $\text{Overlap}_{13} = 0.0\%$ for question definitions. Partial matches ($<1.2\%$) were restricted to common entity names (e.g., "Cobalt Strike", "Mimikatz") rather than specific reasoning chains.

C.3 Testset Slot Guessing (TS-Guessing)

To empirically verify the absence of memorisation, we applied the Testset Slot Guessing protocol. We selected $n = 50$ random questions stratified across difficulty tiers from the benchmark, masked the correct option, and prompted the model to generate the missing answer string zero-shot. The CyberSecEval Malware Analysis benchmark uses a multi-label format with 9 options per question, where the number of correct answers K varies from 1 to 9 [13]. Following the benchmark's baseline computation, the expected accuracy for a random guesser attempting perfect multi-label match is:

$$\begin{aligned} \text{Expected accuracy} &= \sum_{K=1}^{9} p_K \cdot \Pr(\text{perfect} \mid K) \\ &= \sum_{K=1}^{9} \frac{p_K}{9\binom{9}{K}} \approx 0.63\%, \end{aligned} \tag{8}$$

where p_K is the proportion of questions with exactly K correct answers. For single-option guessing, the baseline is $\approx 4.3\%$.

The model achieved a slot-guessing accuracy of 13.8% (7 of 50 questions). While this exceeds the random baseline, it remains far below the performance achieved during normal evaluation with the evidence bundle (35.30%). Critically, the 13.8% accuracy on masked questions reflects the model's general cyber security domain knowledge acquired during pre-training on public threat intelligence, not memorisation of specific benchmark QA pairs. This interpretation is supported by three observations: (1) the model's errors on slot-guessing were semantically plausible alternatives (e.g., confusing related MITRE techniques), not random guesses; (2) performance on samples from the wild collected after the training cutoff (Section 4.6) matches benchmark performance, which would not occur if benchmark-specific memorisation drove accuracy; and (3) the audit found no verbatim overlaps within the available reference corpus (Section C.2).

D Qualitative Case Studies

To address the limitations of multiple-choice benchmarks, we evaluated the hybrid system on malware samples collected from public threat intelligence feeds in January 2026. We curated a set of 12 samples exhibiting novel evasion techniques not represented in CyberSecEval, selecting samples based on three criteria: (1) availability of detailed detonation reports from Hybrid Analysis, (2) use of techniques documented in 2024–2025 threat intelligence (EtherHiding, ClickFix, clipboard injection), and (3) presence of obfuscation patterns known to degrade LLM reasoning [7]. The hybrid system correctly classified 9 of 12 samples (75.0%), compared to 5 of 12 (41.7%) for the single-model baseline (Gemini 3 Pro Preview).

Table 6 presents an overview of the qualitative analysis of the two case studies, which are detailed below.

Table 6. Qualitative Analysis Summary: Single-Model Baseline vs. Hybrid System

Malware Sample	Evasion Technique	Single-Model Failure	Hybrid Intervention	Hybrid System's Verdict
UNC5142 (Case A)	**EtherHiding**	Misclassified as *Cryptojacking/Mining* due to blockchain keywords.	**Phase 1 (Agentic):** Evidence Miner used grep to isolate payload in transaction logs.	**Downloader / Dropper**
Lumma Stealer (Case B)	**ClickFix**	Misclassified as *Credential Phishing Site* based on visual lure text.	**Phase 2 (Debate):** Expert Agent linked clipboard event handlers to LummaC2 chains.	**Lumma Stealer**

Evaluation Methodology. Ground truth labels for all 12 samples were established *prior* to system evaluation using published threat intelligence reports from security vendors (Mandiant, Microsoft MSTIC, Group-IB, Sekoia) that pre-dated our analysis. Each sample's ground truth comprised: (1) malware family classification, (2) primary delivery/evasion technique, and (3) key indicators of compromise (IOCs). To mitigate evaluator bias, we used a blinded protocol: system outputs were anonymised (labelled "System A" and "System B") before correctness assessment, and the evaluator did not know which system produced which output until after all 12 samples were scored. Correctness was assessed by a single evaluator (an author with 3+ years of malware analysis experience) using strict criteria: a classification was marked correct only if it matched the ground truth malware family *and* identified the primary technique; partial matches were scored as incorrect. The complete ground truth labels, anonymised system outputs, and per-sample scoring rationale are provided in the anonymous repository to enable independent verification.

Baseline Comparison Methodology. Both the single-model baseline and the Hybrid System received identical inputs: the complete Hybrid Analysis JSON report and a standardised prompt requesting threat classification and technique identification. The baseline received no tool access or multi-round reasoning, reflecting typical single-pass LLM deployment.

Case A: UNC5142 "EtherHiding" Campaign

Sample Overview. A Hybrid Analysis report detailing the **UNC5142** campaign [37]. This campaign utilises "EtherHiding," a technique where malicious payloads are stored within the *data* field of blockchain smart contracts (specifically Binance Smart Chain) rather than on traditional C2 servers [50]. UNC5142 was active from December 2024 through mid-2025, distributing infostealers including Lumma and Vidar variants.

The Challenge. The report contains extensive blockchain transaction logs and obscure JavaScript that retrieves data from a specific contract address. There are no standard HTTP URLs pointing to a payload, obscuring the infection vector from standard pattern matching.

Single-Model Failure. The single LLM (Gemini 3 Pro Preview) correctly identified the presence of blockchain elements but hallucinated the threat intent. It classified the sample as "Cryptojacking/Mining" software intended to steal CPU resources (confidence: 0.71), missing the actual delivery mechanism. It failed to locate the payload source, stating: *"No direct malware download URL was found in the provided code."*

Hybrid System Success. In Phase 1 (Agentic), the *Evidence Miner* successfully executed `grep` patterns for hexadecimal strings within the transaction logs, isolating the payload data chunk. In Phase 2 (Debate), a critical disagreement occurred in Round 2. The General Agent (Qwen3-4B) initially argued the contract was for "payment processing." The Expert Agent (Foundation-Sec-8B) countered by citing the specific `data` field anomaly, arguing: *"The contract logic does not process tokens; it serves immutable data blobs consistent with EtherHiding infrastructure documented in recent GTIG advisories."* The system correctly classified the sample as a **Downloader/Dropper** (confidence: 0.89) and accurately extracted the BSC contract address serving the payload.

Case B: Lumma Stealer with "ClickFix"

Sample Overview. A Hybrid Analysis report of a **Lumma Stealer** variant [44]. This sample uses the "ClickFix" social engineering tactic [24], using a fake Google Chrome update overlay that tricks users into copying a PowerShell command into their clipboard to "fix" a display error [28,59].

The Challenge. The malicious logic is hidden inside an HTML clipboard event handler (`oncopy`/`onclick`), while the bulk of the report describes benign HTML structure and CSS. The attack relies on the user manually pasting the payload into the Windows "Run" dialog, bypassing standard browser download protections.

Single-Model Failure. The single LLM focused heavily on the visual aspects described in the report (the "Update Chrome" text) and classified it as a "Credential Phishing Site" intended to steal login passwords (confidence: 0.68). It missed the specific PowerShell execution vector entirely.

Hybrid System Success. In Phase 1 (Agentic), the *Evidence Miner* extracted the specific `oncopy` and `onclick` JavaScript event handlers that facilitate the clipboard hijacking. In Phase 2 (Debate), the Expert Agent successfully linked the `powershell -w hidden -enc` command found in the clipboard buffer to characteristics consistent with documented LummaC2 infection chains, citing the Base64-encoded payload structure and `Invoke-WebRequest` patterns typical of 2025 variants. The system correctly identified the threat as **Lumma Stealer** (confidence: 0.92) and flagged "ClickFix" clipboard injection as the initial access vector (MITRE ATT&CK T1059.001, T1204.002).

Table 7. The debate system's overall accuracy (7 rounds, selected based on the optimal trade-off between hard-question gains and easy-question degradation shown in Figure 4) for all pairwise model combinations on the Malware Analysis benchmark. Only the upper triangle is shown due to the symmetry of the debate setting. Rows represent Agent A, columns represent Agent B in the debate. Diagonal entries show self-debate (same model for both agents). Values represent overall accuracy across all 609 questions.

Agent A \ Agent B	Qwen3-0.6B	Llama-3.2-1B	Qwen2.5-1.5B	DeepSeek-R1-1.5B	SmolLM2-1.7B	Phi-3.5-mini-instruct	Qwen3-4B	Gemma-3-4B-IT	Qwen2.5-Coder-7B	Llama-3.1-8B	Ministral-8B	DeepHat-V1-7B	Foundation-Sec-8B
Qwen3-0.6B	16.4%	15.1%	15.8%	15.5%	15.2%	16.2%	16.8%	16.5%	17.1%	17.3%	17.5%	17.8%	18.6%
Llama-3.2-1B		15.6%	15.9%	16.1%	15.8%	16.7%	17.2%	16.9%	17.6%	17.8%	17.9%	18.3%	19.1%
Qwen2.5-1.5B			16.2%	16.4%	16.0%	17.1%	17.6%	17.3%	18.0%	18.2%	18.4%	18.8%	19.5%
DeepSeek-R1-1.5B				16.8%	16.3%	17.5%	18.0%	17.7%	18.4%	18.6%	18.8%	19.2%	19.9%
SmolLM2-1.7B					16.0%	17.0%	17.5%	17.2%	17.8%	18.0%	18.2%	18.6%	19.3%
Phi-3.5-mini-instruct						19.5%	19.8%	19.6%	20.4%	20.6%	20.8%	20.5%	21.2%
Qwen3-4B							20.2%	20.0%	20.7%	20.9%	21.1%	21.1%	**24.13%**
Gemma-3-4B-IT								19.8%	20.5%	20.7%	20.9%	20.8%	21.6%
Qwen2.5-Coder-7B									21.4%	21.8%	22.0%	22.3%	23.2%
Llama-3.1-8B										21.6%	22.2%	22.5%	23.4%
Ministral-8B											23.0%	22.7%	23.6%
DeepHat-V1-7B												16.8%	18.2%
Foundation-Sec-8B													20.0%

E Debate Partner Selection: Full Matrix Analysis

Table 7 reports the full all-pairs debate-partner results (7 rounds) used for the partner-selection analysis in Section 4. To explore the impact of debate partner selection systematically, Table 7 presents results for all pairwise model combinations, treating general-purpose SLMs and cyber-specialised models (DeepHat-V1-7B and Foundation-Sec-8B) uniformly. Notably, the matrix is symmetric: swapping which model acts as Agent A versus Agent B produces identical accuracy, indicating that debate outcomes are independent of role assignment. Beyond this symmetry, the matrix reveals three critical patterns. First, *complementary expertise matters*: the best-performing configurations pair strong general-purpose models with cyber-specialised experts (e.g., Qwen3-4B with Foundation-Sec-8B achieves 24.13%, Ministral-8B with Foundation-Sec-8B achieves 23.6%, Llama-3.1-8B with Foundation-Sec-8B reaches 23.4%), consistently outperforming both general-plus-general debates (Ministral-8B with Llama-3.1-8B yields 22.2%) and cyber-plus-cyber debates (Foundation-Sec-8B with DeepHat-V1-7B yields only 18.2%). Second, larger parameter imbalance coincided with lower performance in the tested pairs: Qwen3-0.6B with Foundation-Sec-8B achieved 18.6%, whereas stronger mid-sized models paired with the same expert performed better. Because model size covaries with solo accuracy and specialisation, this is an observed association rather than validation of a 4× threshold. Third, *self-debates underperform* except for the largest models: Ministral-8B de-

bating with itself achieves 23.0%, only slightly below its cross-model debates, while Qwen3-0.6B self-debate stalls at 16.4%, barely exceeding its single-model baseline of 11.05%.

References

1. Akhtar, M.S., Feng, T.: Evaluation of machine learning algorithms for malware detection. Sensors **23**(2), 946:1–946:17 (2023). https://doi.org/10.3390/s23020946
2. Al-Karaki, J., Khan, M.A.Z., Omar, M.: Exploring LLMs for malware detection: Review, framework design, and countermeasure approaches. Preprint arXiv:2409.07587 [cs.CR] (2024). https://doi.org/10.48550/arXiv.2409.07587
3. Anthropic: System card: Claude Opus 4.5. Online document (2025), https://www-cdn.anthropic.com/bf10f64990cfda0ba858290be7b8cc6317685f47.pdf
4. Aslan, Ö., Ozkan-okay, M., Gupta, D.: Intelligent behavior-based malware detection system on cloud computing environment. IEEE Access **9**, 83252–83271 (2021). https://doi.org/10.1109/access.2021.3087316
5. Belaoued, M., Derhab, A., Mazouzi, S., Khan, F.A.: MACoMal: A multi-agent based collaborative mechanism for anti-malware assistance. IEEE Access **8**, 14329–14343 (2020). https://doi.org/10.1109/access.2020.2966321
6. Belcak, P., Heinrich, G., Diao, S., Fu, Y., Dong, X., Muralidharan, S., Lin, Y.C., Molchanov, P.: Small language models are the future of agentic AI. Preprint arXiv:2506.02153 (2025). https://doi.org/10.48550/arXiv.2506.02153
7. Böke, E., Torka, S.: "Digital Camouflage": The LLVM challenge in LLM-based malware detection. Journal of Systems and Software pp. 112646:1–112646:11 (2025)
8. Cai, W., Jiang, J., Wang, F., Tang, J., Kim, S., Huang, J.: A survey on mixture of experts in large language models. IEEE Trans. on Knowledge and Data Engineering **37**, 3896–3915 (2024). https://doi.org/10.1109/tkde.2025.3554028
9. Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., Oprea, A., Raffel, C.: Extracting training data from large language models. In: Proceedings of the 30th USENIX Security Symposium. pp. 2633–2650. USENIX Association (2021), https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting
10. Catak, F.O., Yazi, A.F., Elezaj, O., Ahmed, J.: Deep learning based sequential model for malware analysis using windows exe API calls. PeerJ Computer Science **6**, e285:1–e285:23 (2020). https://doi.org/10.7717/peerj-cs.285
11. Chen, J.C.Y., Saha, S., Bansal, M.: ReConcile: Round-table conference improves reasoning via consensus among diverse LLMs. Preprint arXiv:2309.13007 [cs.CL] (2023). https://doi.org/10.48550/arXiv.2309.13007
12. CrowdStrike: Hybrid Analysis: Free automated malware analysis service. Website (2024), https://www.hybrid-analysis.com/, accessed 2025-12-01
13. Deason, L., Bali, A., Bejean, C., Bolocan, D., Crnkovich, J., Croitoru, I., Durai, K., Midler, C., Miron, C., Molnar, D., Moon, B., Ostarcevic, B., Peltea, A., Rosenberg, M., Sandu, C., Saputkin, A., Shah, S., Stan, D., Szocs, E., Wan, S., Whitman, S., Krasser, S., Saxe, J.: CyberSOCEval: Benchmarking LLMs capabilities for malware analysis and threat intelligence reasoning. Preprint arXiv:2509.20166 [cs.CR] (2025). https://doi.org/10.48550/arXiv.2509.20166
14. DeepHat: DeepHat-V1-7B. LLM repo (2024), https://huggingface.co/DeepHat/DeepHat-V1-7B, accessed 2025-12-01

15. DeepSeek: DeepSeek-R1-Distill-Qwen-1.5B. LLM repo (2024), https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B, accessed 2025-12-01
16. DeepSeek-AI Team: DeepSeek-V3.2: Pushing the frontier of open large language models. Preprint arXiv:2512.02556 [cs.CL] (2025). https://doi.org/10.48550/arXiv.2512.02556
17. Du, Y., Li, S., Torralba, A., Tenenbaum, J.B., Mordatch, I.: Improving factuality and reasoning in language models through multiagent debate. In: Proceedings of the 41st International Conference on Machine Learning. pp. 8465–8479. PMLR (2025), https://raw.githubusercontent.com/mlresearch/v235/main/assets/du24e/du24e.pdf
18. Fan, M., Wei, W., Xie, X., Liu, Y., Guan, X., Liu, T.: Can we trust your explanations? sanity checks for interpreters in Android malware analysis. IEEE Transactions on Information Forensics and Security **16**, 838–853 (2020). https://doi.org/10.1109/tifs.2020.3021924
19. fdtn-ai: Foundation-Sec-8B-Instruct. LLM repo (2024), https://huggingface.co/fdtn-ai/Foundation-Sec-8B-Instruct, accessed 2025-12-01
20. Gemini Team, Google: Gemini: A family of highly capable multimodal models. Preprint arXiv:2312.11805v5 [cs.CL] (2025), https://arxiv.org/abs/2312.11805v5
21. Giarimpampa, D., Meier, R., Bissyande, T.F., Lenders, V., Klein, J.: Exploring the role of artificial intelligence in enhancing security operations: A systematic review. ACM Computing Surveys **58**(3), 67:1–67:38 (2025). https://doi.org/10.1145/3747587
22. Golchin, S., Surdeanu, M.: Time travel in LLMs: Tracing data contamination in large language models. Preprint arXiv:2308.08493v3 [cs.CL] (2024), https://arxiv.org/abs/2308.08493v3
23. Google DeepMind: Gemma 3 4B IT. LLM repo (2024), https://huggingface.co/google/gemma-3-4b-it, accessed 2025-12-01
24. Group-IB Threat Intelligence: ClickFix: The social engineering technique hackers use to manipulate victims. Web page (2025), https://www.group-ib.com/blog/clickfix-the-social-engineering-technique-hackers-use-to-manipulate-victims/, accessed: 2026-02-04
25. Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J.C.: Large language models for software engineering: A systematic literature review. ACM Transactions on Software Engineering and Methodology **33**, 220:1–220:79 (2023). https://doi.org/10.1145/3695988
26. Hugging Face: all-MiniLM-L6-v2. Model repository (2021), https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2, accessed 2025-12-01
27. Hugging Face: SmolLM2-1.7B. LLM repo (2024), https://huggingface.co/HuggingFaceTB/SmolLM2-1.7B, accessed 2025-12-01
28. Huntress Threat Ops: ClickFix gets creative: Malware buried in images. Web page (2025), https://www.huntress.com/blog/clickfix-malware-buried-in-images, accessed: 2026-02-04
29. Kuppa, A., Aouad, L., Le-Khac, N.A.: Linking CVE's to MITRE ATT&CK techniques. In: Proceedings of the 16th International Conference on Availability, Reliability and Security. pp. 21:1–21:12. ACM (2021). https://doi.org/10.1145/3465481.3465758
30. Leon, M.: GPT-5 and open-weight large language models: Advances in reasoning, transparency, and control. Information Systems pp. 102620:1–102620:9 (2025). https://doi.org/10.1016/j.is.2025.102620

31. Levi, M., Ohayon, D., Blobstein, A., Sagi, R., Molloy, I., Allouche, Y.: Toward cybersecurity-expert small language models. Preprint arXiv:2510.14113 [cs.CL] (2025). https://doi.org/10.48550/arXiv.2510.14113
32. Li, M.Q., Fung, B.C.M.: Security concerns for large language models: A survey. Journal of Information Security and Applications **95**, 104284:1–104284:18 (2025). https://doi.org/10.1016/j.jisa.2025.104284
33. Lin, F., Kim, D.J., Chen, T.H.P.: SOEN-101: Code Generation by Emulating Software Process Models Using Large Language Model Agents, pp. 1527–1539. IEEE (2025). https://doi.org/10.1109/ICSE55347.2025.00140
34. Liu, Z., Zhang, Y., Li, P., Liu, Y., Yang, D.: A dynamic LLM-powered agent network for task-oriented agent collaboration. Preprint arXiv:2310.02170 [cs.CL] (2023). https://doi.org/10.48550/arXiv.2310.02170
35. Lu, Z., Li, X., Cai, D., Yi, R., Liu, F., Zhang, X., Lane, N.D., Xu, M.: Small language models: Survey, measurements, and insights. Preprint arXiv:2409.15790 [cs.CL] (2025). https://doi.org/10.48550/arXiv.2409.15790
36. Mahmoud, R., Anagnostopoulos, M., Pastrana, S., Pedersen, J.M.: Redefining malware sandboxing: Enhancing analysis through sysmon and ELK integration. IEEE Access **12**, 68624–68636 (2024). https://doi.org/10.1109/access.2024.3400167
37. Mandiant Threat Intelligence: New group on the block: UNC5142 leverages EtherHiding to distribute malware, Google Cloud Threat Intelligence Blog. Web page (2025), https://cloud.google.com/blog/topics/threat-intelligence/unc5142-etherhiding-distribute-malware, accessed: 2026-02-04
38. Manthena, H., Shajarian, S., Kimmell, J., Abdelsalam, M., Khorsandroo, S., Gupta, M.: Explainable artificial intelligence (XAI) for malware analysis: A survey of techniques, applications, and open challenges. IEEE Access **13**, 61611–61640 (2024). https://doi.org/10.1109/access.2025.3555926
39. Mat, N., Jamil, N., Yusoff, Y., Kiah, M.L.M.: A systematic literature review on advanced persistent threat behaviors and its detection strategy. Journal of Cybersecurity **10**(1), tyad023:1–tyad023:18 (2024). https://doi.org/10.1093/cybsec/tyad023
40. Meta AI: Llama 3.1 8B Instruct. LLM repo (2024), https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct, accessed 2025-12-01
41. Meta AI: Llama 3.2 1B. LLM repo (2024), https://huggingface.co/meta-llama/Llama-3.2-1B, accessed 2025-12-01
42. Meta Team: The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. Web page (2025), https://ai.meta.com/blog/llama-4-multimodal-intelligence
43. Microsoft: Phi-3.5 Mini Instruct. LLM repo (2024), https://huggingface.co/microsoft/Phi-3.5-mini-instruct, accessed 2025-12-01
44. Microsoft Threat Intelligence: Lumma Stealer: Breaking down the delivery techniques and capabilities of a prolific infostealer. Web page (2025), https://www.microsoft.com/en-us/security/blog/2025/05/21/lumma-stealer-breaking-down-the-delivery-techniques-and-capabilities-of-a-prolific-infostealer, accessed: 2026-02-04
45. Mienye, I.D., Sun, Y.: A survey of ensemble learning: Concepts, algorithms, applications, and prospects. IEEE Access **10**, 99129–99149 (2022). https://doi.org/10.1109/access.2022.3207287
46. Mistral AI: Ministral-8B-Instruct-2410. LLM repo (2024), https://huggingface.co/mistralai/Ministral-8B-Instruct-2410, accessed 2025-12-01
47. OpenAI Team: GPT-4 technical report. Preprint arXiv:2303.08774v6 [cs.CL] (2024), https://arxiv.org/abs/2303.08774v6

48. Or-Meir, O., Nissim, N., Elovici, Y., Rokach, L.: Dynamic malware analysis in the modern era—a state of the art survey. ACM Computing Surveys **52**, 88:1–88:48 (2019). https://doi.org/10.1145/3329786
49. Patsakis, C., Casino, F., Lykousas, N.: Assessing LLMs in malicious code deobfuscation of real-world malware campaigns. Expert Systems with Applications **256**, 124912:1–124912:13 (2024). https://doi.org/10.1016/j.eswa.2024.124912
50. Picus Security: EtherHiding: How Web3 infrastructure enables stealthy malware distribution. Web page (2025), https://www.picussecurity.com/resource/blog/etherhiding-how-web3-infrastructure-enables-stealthy-malware-distribution, accessed: 2026-02-04
51. Qaisar, Z.H., Almotiri, S.H., Al Ghamdi, M.A., Nagra, A.A., Ali, G.: A scalable and efficient multi-agent architecture for malware protection in data sharing over mobile cloud. IEEE Access **9**, 76248–76259 (2021). https://doi.org/10.1109/access.2021.3067284
52. Qwen Team: Qwen2.5 1.5B Instruct. LLM repo (2024), https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct, accessed 2025-12-01
53. Qwen Team: Qwen2.5-Coder 7B Instruct. LLM repo (2024), https://huggingface.co/Qwen/Qwen2.5-Coder-7B-Instruct, accessed 2025-12-01
54. Qwen Team: Qwen3-0.6B. LLM repo (2024), https://huggingface.co/Qwen/Qwen3-0.6B, accessed 2025-12-01
55. Qwen Team: Qwen3-4B. LLM repo (2025), https://huggingface.co/Qwen/Qwen3-4B, accessed 2025-12-01
56. Raza, M., Jahangir, Z., Riaz, M.B., Saeed, M.J., Sattar, M.A.: Industrial applications of large language models. Scientific Reports **15**, 13755:1–13755:23 (2025). https://doi.org/10.1038/s41598-025-98483-1
57. Sagi, O., Rokach, L.: Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8**(4), e1249:1–e1249:18 (2018). https://doi.org/10.1002/widm.1249
58. Saqib, M., Mahdavifar, S., Fung, B.C.M., Charland, P.: A comprehensive analysis of explainable AI for malware hunting. ACM Computing Surveys **56**, 314:1–314:40 (2024). https://doi.org/10.1145/3677374
59. Sekoia.io Threat & Detection Research: Meet IClickFix: a widespread WordPress-targeting framework using the ClickFix tactic. Web page (2026), https://blog.sekoia.io/meet-iclickfix-a-widespread-wordpress-targeting-framework-using-the-clickfix-tactic/, accessed: 2026-02-04
60. Stokes, P.: Building an adversarial consensus engine: Multi-agent LLMs for automated malware analysis. SentinelOne Labs web page (2026), https://www.sentinelone.com/labs/building-an-adversarial-consensus-engine-multi-agent-llms-for-automated-malware-analysis/, accessed: 2026-07-14
61. Strom, B.E., Applebaum, A., Miller, D.P., Nickels, K.C., Pennington, A.G., Thomas, C.B.: MITRE ATT&CK: Design and philosophy. Tech. rep., The MITRE Corporation (2018), https://www.mitre.org/sites/default/files/2021-11/prs-19-01075-28-mitre-attack-design-and-philosophy.pdf
62. Tran, K.T., Dao, D., Nguyen, M.D., Pham, Q.V., O'Sullivan, B., Nguyen, H.D.: Multi-agent collaboration mechanisms: A survey of LLMs. Preprint arXiv:2501.06322 [cs.AI] (2025). https://doi.org/10.48550/arXiv.2501.06322
63. Ullah, S., Balasubramanian, P., Guo, W., Burnett, A., Pearce, H., Kruegel, C., Vigna, G., Stringhini, G.: From CVE entries to verifiable exploits: An automated multi-agent framework for reproducing CVEs. Preprint arXiv:2509.01835 [cs.CR] (2025). https://doi.org/10.48550/arXiv.2509.01835

64. Uysal, D.T., Yoo, P.D., Taha, K., Yeun, C., Damiani, E.: A multi-label visualisation approach for malware behaviour analysis. Scientific Reports **15**, 37979:1–37979:21 (2025). https://doi.org/10.1038/s41598-025-21848-z
65. Wan, S., Nikolaidis, C., Song, D., Molnar, D., Crnkovich, J., Grace, J., Bhatt, M., Chennabasappa, S., Whitman, S., Ding, S., Ionescu, V., Li, Y., Saxe, J.: CYBERSECEVAL 3: Advancing the evaluation of cybersecurity risks and capabilities in large language models. Preprint arXiv:2408.01605 [cs.CR] (2024). https://doi.org/10.48550/arXiv.2408.01605
66. Wang, J., Wang, J., Athiwaratkun, B., Zhang, C., Zou, J.: Mixture-of-agents enhances large language model capabilities. Preprint arXiv:2406.04692 [cs.CL] (2024). https://doi.org/10.48550/arXiv.2406.04692
67. Wang, Z., Bukharin, A., Delalleau, O., Egert, D., Shen, G., Zeng, J., Kuchaiev, O., Dong, Y.: HelpSteer2-preference: Complementing ratings with preferences. Preprint arXiv:2410.01257 [cs.LG] (2024). https://doi.org/10.48550/arXiv.2410.01257
68. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A.H., White, R.W., Burger, D., Wang, C.: AutoGen: Enabling next-gen LLM applications via multi-agent conversation. Preprint arXiv:2308.08155 [cs.AI]. https://doi.org/10.48550/arXiv.2308.08155
69. Xu, H., Wang, S., Li, N., Wang, K., Zhao, Y., Chen, K., Yu, T., Liu, Y., Wang, H.: Large language models for cyber security: A systematic literature review. ACM Transactions on Software Engineering and Methodology (2025). https://doi.org/10.1145/3769676
70. Yigit, Y., Ferrag, M., Ghanem, M.C., Sarker, I.H., Maglaras, L.A., Chrysoulas, C., Moradpoor, N., Tihanyi, N., Janicke, H.: Generative AI and LLMs for critical infrastructure protection: Evaluation benchmarks, agentic AI, challenges, and opportunities. Sensors **25**(6), 1666:1–1666:40 (2025). https://doi.org/10.3390/s25061666
71. Yu, J., Xie, X., Hu, Q., Ma, Y., Zhao, Z.: Chimera: Harnessing multi-agent LLMs for automatic insider threat simulation. In: Proceedings of the Network and Distributed System Security Symposium (NDSS). Internet Society (2026). https://doi.org/10.14722/ndss.2026.240375
72. Zhang, Q., Liu, Z., Pan, S., Wang, C.: The rise of small language models. IEEE Intelligent Systems **40**, 30–37 (2025). https://doi.org/10.1109/mis.2024.3517792
73. Zhang, W., Zeng, L., Xiao, Y., Li, Y., Cui, C., Zhao, Y., Hu, R., Liu, Y., Zhou, Y., An, B.: AgentOrchestra: Orchestrating multi-agent intelligence with the tool-environment-agent (TEA) protocol. arXiv:2506.12508v5 [cs.AI] (2026), https://arxiv.org/abs/2506.12508v5
74. Zhou, Y., Chen, Y.: Adaptive heterogeneous multi-agent debate for enhanced educational and factual reasoning in large language models. Journal of King Saud University Computer and Information Sciences **37**(10), 330:1–330:19 (2025). https://doi.org/10.1007/s44443-025-00353-3
75. Çetin, O., Ekmekcioglu, E., Arief, B., Hernandez-Castro, J.: An empirical evaluation of large language models in static code analysis for PHP vulnerability detection. Journal of Universal Computer Science **30**(9), 1163–1183 (2024). https://doi.org/10.3897/jucs.134739