Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Multi-Agent Framework Based on Local LLMs for Malware Detection and Behavior Analysis in the ICS/OT Environments of Critical Infrastructure

Sunghun Jang
Ajou University
MyoungRak Lee
Ajou University
Taeshik Shon
tsshon@ajou.ac.kr

Ajou University

Article

Keywords: Cyber-Physical Power and Energy Systems, Critical Infrastructure Protection, ICS/OT Security, Local LLM, Heterogeneous Multi-Agent System, Malware Detection

Posted Date: November 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11587404/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Power and energy facilities depend on cyber-physical control systems whose networks restrict external connectivity, so malware analysis must run on local resources. We present the Multi-Agent Cybersecurity Framework (MACF), which assigns heterogeneous local large language models (LLMs) to static-feature interpretation, behavior inference, offline threat-intelligence retrieval and coordination. MACF targets the Windows portable-executable (PE) malware that reaches the IT-side hosts of power-generation and substation sites, the entry path into their ICS/OT environments.

The agents exchange structured evidence through the Query-Analysis Communication Protocol (QACP); the Adjudication and Cross-validation Mechanism (ACM) triggers re-verification when verdicts conflict or confidence is low; and the Role Fitness Score (RFS) assigns models to roles and weights their verdicts. On 1,000 EMBER 2018-based specimens (800 malicious, 200 benign), MACF achieved a 96.0% detection rate, 8.5% false-positive rate, 96.9% F1 score, 83.1% family-classification accuracy and 75.4% ATT&CK technique match rate, exceeding the best single-LLM baseline by 9.2 percentage points in detection and a homogeneous four-role configuration by 7.4 points. All four quantized models remained resident on a single 128-GB unified-memory server, with a mean analysis time of 61.7 s per specimen. The results show that fully offline, explainable multi-agent malware triage is practical on one server.

1. Introduction

1.1 Background and Motivation

Large language models (LLMs) are increasingly applied to malware analysis, incident response, log interpretation and threat-intelligence summarization [1, 2]. Because they can read structured technical evidence and explain it in natural language, locally hosted LLMs are a promising basis for analyst-oriented malware triage.

Malware has repeatedly reached the industrial control systems of power and energy infrastructure, as Stuxnet [3], TRITON/TRISIS [4] and the attacks on the Ukrainian power grid [5] demonstrate. In each case, Windows hosts on the IT side of the facility, such as engineering workstations and file-transfer paths, served as entry or staging points before the intrusion reached operational equipment. Triage of Windows PE files on these hosts is therefore a practical first line of defense for operators of power-generation, transmission and substation facilities, and it is the problem addressed in this paper.

ICS/OT operations prioritize availability and often limit disruptive security maintenance [6]. Network separation and restrictions on external services also rule out cloud sandboxes and live threat-intelligence feeds. NIST SP 800 – 82 Revision 3 [6] and the IEC 62443 series [7] set out security requirements for such environments, and the EU NIS2 Directive [8] strengthens incident-reporting obligations for essential entities, including energy operators. Together, these constraints call for an on-site analysis workflow that processes suspicious files without any external API calls.

MACF meets this need by assigning different local LLMs to static analysis, behavior inference, threat-intelligence retrieval and coordination. The agents exchange structured evidence, and re-verification is invoked when their verdicts conflict or their confidence falls below a configured threshold.

1.2 Research Objectives and Contributions

This study makes three contributions. First, it designs a fully local multi-agent pipeline for PE-malware triage that combines static features, inferred behavior and offline threat intelligence. Second, it introduces three coordination mechanisms: QACP for structured evidence exchange, ACM for conditional re-verification and RFS for model-to-role assignment and verdict weighting. Third, it evaluates the pipeline on 1,000 EMBER 2018-based specimens, reporting detection, interpretation, ablation and resource results against four single-LLM baselines and a homogeneous four-role configuration.

Section 2 reviews related work, Section 3 describes MACF, Section 4 presents the experiments and results, Section 5 illustrates two ICS/OT operating scenarios, and Section 6 concludes with limitations and future work.

2. Related Work

2.1 Single LLM-Based Cybersecurity Research

Large language models (LLMs) have been investigated for various cybersecurity tasks, including malware analysis, vulnerability detection, security log interpretation, and threat intelligence processing [9]. Their ability to interpret structured technical information and generate natural-language explanations enables analysts to examine security evidence within a unified analytical workflow. In malware analysis, LLMs can process information derived from portable executable (PE) files, including metadata, imported functions, strings, and structural indicators, to support malicious-file classification and the interpretation of potential attack behavior.

However, LLM-based security analysis presents several reliability challenges. Model outputs may vary across repeated queries, and generated interpretations may contain unsupported conclusions or inaccurate associations between observed evidence and malicious behavior [10]. These limitations are particularly relevant to malware analysis, where incorrect judgments can affect subsequent incident-response decisions. Although a single LLM can process multiple types of evidence, its conclusions are not independently verified merely by combining those inputs.

Furthermore, cloud-based LLM services may be unsuitable for environments where malware samples and internal security information cannot be transmitted externally. Local deployment addresses this connectivity constraint but requires consideration of available computing resources and model capabilities.

These considerations motivate local LLMs with explicit evidence management and conditional verification, the approach taken in MACF (Section 3).

2.2 Multi-Agent LLM Systems

Multi-agent LLM systems distribute tasks such as planning, reasoning, verification, and reporting among multiple agents that collaborate through structured interactions. Representative approaches include AutoGen [11], which supports conversational coordination among agents; MetaGPT [12], which organizes specialized roles through structured workflows; and CAMEL [13], which investigates role-based communication between language-model agents. Generative agents [14] have also demonstrated how language models can simulate autonomous behavior and interactions, while multi-agent debate [15] has explored the use of iterative discussion to improve reasoning and factual consistency.

These studies establish the potential of role specialization and inter-agent communication for complex tasks. In cybersecurity, Liu [16] investigated LLM-based multi-agent collaboration in simulated incident-response scenarios using centralized, decentralized, and hybrid organizational structures. This research demonstrates the relevance of collaboration design to security decision-making and provides a foundation for examining agent coordination in specialized security applications.

However, applying multi-agent collaboration to malware analysis introduces additional requirements. Static feature interpretation, behavior inference, and threat intelligence retrieval produce different forms of evidence that must be associated with the same specimen. Their conclusions may also conflict or exhibit different levels of confidence. Consequently, an analysis framework requires mechanisms for maintaining shared context, exchanging structured evidence, identifying inconsistent judgments, and determining when additional verification is necessary. Multiple agent interactions also introduce inference overhead, making conditional re-verification relevant to resource-constrained deployments.

MACF adapts this role-based paradigm to local malware triage; its coordination mechanisms are detailed in Section 3.

2.3 Malware Detection Techniques

Malware detection techniques can be broadly categorized into static and dynamic analysis approaches. Static analysis examines executable-file characteristics without executing the program, using information such as PE headers, imported functions, strings, section entropy, and byte sequences. Machine-learning approaches use these features to distinguish malicious and benign files. For example, MalConv [17] applies deep learning directly to executable byte sequences, demonstrating an alternative to manually engineered static feature representations.

Dynamic analysis examines program behavior during execution, including system calls, process activity, file operations, and network communication [18]. These observations provide behavioral evidence that may not be directly available from static features. However, dynamic analysis requires controlled execution environments and may be affected by malware that modifies its behavior in response to the analysis environment. In addition, adversarial malware research has demonstrated that modifications to

executable files or their feature representations can affect the predictions of learning-based detectors [19].

Static and dynamic analysis therefore provide different perspectives on malicious activity. Conventional classifiers offer established approaches to binary malware detection, while additional analytical processes may be required to associate classification results with malware families, attack techniques, and supporting evidence. Integrating these forms of interpretation is particularly relevant to security analysts who must understand the basis of a detection result.

MACF is complementary to these techniques: it works from structured static features and adds malware-family, ATT&CK, and evidence-level interpretation, without replacing conventional classifiers or performing dynamic execution (Section 3).

2.4 LLM Operation in Air-Gapped and Edge Environments and Quantization

Operating large language models (LLMs) in air-gapped and resource-constrained environments requires efficient memory management, inference optimization, and locally accessible knowledge resources. Model quantization reduces memory consumption by representing model parameters at lower numerical precision. Representative approaches include GPTQ [20], AWQ [21], SmoothQuant [22], and LLM.int8() [23]. These techniques provide a foundation for deploying large models under hardware constraints. In addition, vLLM [24] supports efficient inference serving through optimized memory management, while retrieval-augmented generation (RAG) [25] enables models to incorporate information from locally maintained knowledge repositories without relying on external services.

However, operating multiple LLMs on a single server introduces additional challenges. Model residency, inference latency, and memory requirements must be considered together, particularly when agents perform sequential analysis and conditional re-verification. Furthermore, offline knowledge repositories require controlled updates to maintain the relevance and traceability of threat information.

MACF combines these techniques so that four quantized models and an offline retrieval component run on a single server; Section 4.7 reports the measured memory and latency.

2.5 LLM-Based ICS/OT Security Research

Security research in industrial control system (ICS) and operational technology (OT) environments has addressed anomaly detection, industrial process monitoring, and the identification of malicious activities affecting control systems. For example, machine-learning approaches have been investigated for detecting cyberattacks from industrial process data [26]. Such methods provide important capabilities for identifying abnormal operational behavior within industrial environments.

However, ICS/OT security also involves threats to Windows-based engineering workstations, human-machine interfaces, and other supporting systems. These assets may be exposed to malicious files

introduced through removable media or connections with business networks. Consequently, file-based malware detection and interpretation constitute a complementary security capability alongside industrial process and protocol monitoring.

LLM-based analysis can support this capability by interpreting structured malware features, inferring potential malicious behavior, and associating the available evidence with threat intelligence. Nevertheless, deployment in restricted environments requires local inference and access to approved offline knowledge resources.

MACF targets this complementary layer; its evaluation is limited to Windows PE features and does not cover PLC firmware, industrial protocols, or process anomalies (Section 3.1).

2.6 Differentiation from Existing Research

The main distinction of MACF lies in the integration of heterogeneous local LLMs, role-specialized analysis, and evidence-based cross-validation within a fully offline architecture. Rather than relying on a single model to perform all analytical functions, MACF assigns different models to coordination, static analysis, behavior inference, and threat intelligence according to their Role Fitness Scores (RFS). This configuration enables the framework to utilize the measured strengths of individual models for specific analytical tasks.

MACF further incorporates two coordination mechanisms. The Query-Analysis Communication Protocol (QACP) standardizes the exchange of verdicts, confidence scores, and supporting evidence among agents. The Adjudication and Cross-validation Mechanism (ACM) initiates additional analysis when agent verdicts conflict, confidence is insufficient, or the reported evidence is inconsistent. These mechanisms establish an explicit process for evidence exchange, re-verification, and final verdict aggregation.

Another characteristic is its deployment architecture. All models and the threat-intelligence knowledge repository operate locally without external inference APIs or real-time cloud services. Through model quantization and sequential inference, MACF accommodates four role-assigned LLMs on a single server with unified memory.

Unlike approaches focused solely on binary malware classification, MACF integrates malicious-file detection with malware-family interpretation, ATT&CK technique mapping, and structured evidence reporting. Its experimental evaluation compares single-LLM baselines, a homogeneous multi-agent configuration, and component-level ablations to examine the performance of the integrated framework.

The contribution of MACF is therefore an integrated, locally deployable malware-analysis workflow that combines heterogeneous model assignment with structured evidence exchange and conditional cross-validation. Its comparative evaluation focuses on LLM-based configurations and does not establish superiority over conventional static malware classifiers or under matched inference budgets.

3. Proposed MACF Framework

3.1 Operational Environment and Analysis Scope

The proposed MACF is designed for malware detection and analysis in the restricted network environments of power, energy and other national critical infrastructure. Industrial control system (ICS) and operational technology (OT) environments commonly include Windows-based engineering workstations, human-machine interfaces (HMIs), and supporting servers that may be exposed to malware introduced through removable media or interconnected IT networks. Compromise of these systems can affect the security and continuity of industrial operations.

These environments prioritize availability and operational continuity, while external connectivity and the use of cloud-based security services may be restricted. Consequently, suspicious files must often be analyzed using locally available computing resources and threat intelligence. MACF addresses these operational requirements by integrating heterogeneous local LLMs for static analysis, behavior inference, and offline threat intelligence within a single-server architecture.

The experimental scope of this study focuses on Windows portable executable (PE) malware represented by structured static features. The framework classifies suspicious specimens, identifies associated malware families, and recommends relevant MITRE ATT&CK techniques based on the available evidence. Behavior analysis refers to inference from static features rather than direct observation of malware execution. Direct analysis of industrial protocol traffic, PLC firmware, and physical process anomalies is outside the scope of the present evaluation.

3.2 Overall Architecture

Figure 1 shows one Coordinator (CO) and three analysis agents. The CO sequences analysis requests and combines agent outputs into a verdict and report; it orchestrates the workflow and composes the final report but does not cast an evidence vote, so Eq. (2) excludes it from the aggregation. The static agent examines structured PE attributes, including imports, strings, entropy, and structural indicators. The behavior agent infers expected activity and ATT&CK associations from static evidence; it does not observe execution. The threat-intelligence agent retrieves offline family, technique, and incident information. QACP carries agent evidence, while ACM initiates conditional re-verification when initial verdicts conflict, confidence is low, or evidence is inconsistent.

Supplementary Figure S1 divides MACF into three layers. The Knowledge Layer stores offline ATT&CK descriptions, malware-family information, and incident references for retrieval. The Model Layer contains the four role-assigned local LLMs and their RFS values. The Orchestration Layer contains the Coordinator, QACP, and ACM, which govern evidence exchange, re-verification, and final aggregation.

3.3 Agent Role Assignment and Role Fitness Score (RFS)

MACF uses four local LLMs assigned to the Coordinator, static-analysis, behavior-inference, and threat-intelligence roles. Let M be the candidate-model set and R the role set. For each model-role pair (m, r), Eq. (1) defines a Role Fitness Score (RFS) as a weighted sum of measured role accuracy, response consistency, and inference efficiency.

$$RFS(m, r) = w_A \cdot A(m, r) + w_C \cdot C(m, r) + w_E \cdot E(m, r),\ w_A + w_C + w_E = 1 \quad (1)$$

Equations (1a)–(1c) define the three components, each normalized to [0, 1].

$$A(m, r) = \text{correct}_r / \text{total}_r \quad (1a)$$

where correct_r and total_r are the numbers of correct and total role-specific validation outputs for role r, respectively.

$$C(m, r) = [\Sigma_{i<j} 1[y_i \equiv y_j)] / [N_r(N_r - 1)/2] \quad (1b)$$

where N_r is the number of repeated queries with identical inputs for role r; $1[\cdot]$ is the indicator function; and y_i is the i-th response. The denominator counts all distinct response pairs.

$$E(m, r) = 1 - t(m, r) / t_{\max} \quad (1c)$$

where $t(m, r)$ is the mean inference latency of model m for role r, and t_{max} is the maximum latency observed among all candidate models for the same evaluation setting. Higher values indicate greater efficiency.

In Eq. (1), w_A, w_C, and w_E are nonnegative component weights satisfying $w_A + w_C + w_E = 1$. A larger RFS(m, r) therefore indicates greater suitability of model m for role r under the selected balance of accuracy, stability, and latency. The model with the largest score for a role is selected subject to the deployment constraints and the one-role-per-agent assignment used in this study.

Supplementary Table S1 reports the RFS values and the resulting role assignments used in this implementation.

Accordingly, EXAONE-Deep 32B [27] serves as the Coordinator, Qwen2.5-Coder 32B [28] as the static agent, Qwen3 32B [29] as the behavior agent, and Llama-3.3 70B [30] with offline RAG as the threat-intelligence agent; each is the highest-scoring model for its role. All agents use role-specific prompts and structured outputs. The RFS values were used for model-to-role assignment and evidence aggregation; because the component weights and repetition counts required to reconstruct these values were not retained, the reported RFS matrix cannot be independently reproduced from the information provided.

3.4 Query-Analysis Communication Protocol (QACP)

QACP keeps context consistent across agents and standardizes their outputs. As shown in Supplementary Figure S2, each message comprises (i) a system component specifying the role and

output schema, (ii) a context component containing the sample hash, metadata and summaries of earlier analyses, and (iii) a query component containing the analysis request for the current stage. Because every agent shares the same sample identifier and accumulated evidence, later agents build on earlier findings instead of answering in isolation, as they would with independent parallel calls.

Every QACP response contains the fields verdict, confidence, evidence, family and technique tags. This shared schema lets the Coordinator compare outputs directly and apply the aggregation rule, and the numerical confidence provides the input to ACM's re-verification trigger.

3.5 Adjudication and Cross-validation Mechanism (ACM)

ACM resolves disagreement among agents in four stages (Fig. 2): independent initial analysis, discrepancy detection, evidence exchange and final consensus. It initiates re-analysis on any of three triggers: (i) the static and behavior agents return conflicting binary verdicts; (ii) they agree, but at least one reported confidence is below the re-verification threshold τ; or (iii) structural indicators, such as packing, contradict the inferred behavior tags. ACM stops when the discrepancy is resolved or after two iterations. The Coordinator then aggregates the evidence agents with Eq. (2) and applies the final-verdict rule in Eq. (3).

$$c^{*} = [\Sigma_{a \in \mathcal{A}_v} \mathrm{RFS}(m_a, r_a) \cdot c_a] / [\Sigma_{a \in \mathcal{A}_v} \mathrm{RFS}(m_a, r_a)] \quad (2)$$

In Eq. (2), $\mathcal{A}_v$ is the set of evidence-producing agents participating in the current verdict, excluding the Coordinator. RFS(m$_a$, r$_a$) is the Role Fitness Score of agent a, where m$_a$ denotes its assigned model and r$_a$ its role. Each agent reports a binary verdict and a confidence q$_a$ ∈ [0.5, 1] in that verdict. QACP maps this value to the common malicious-class scale as c$_a$ = q$_a$ for a malicious verdict and c$_a$ = 1 − q$_a$ for a benign verdict. The denominator normalizes the RFS weights so that c* ∈ [0, 1] represents the RFS-weighted malicious-class support. Because the agent scores are not probabilistically calibrated, c* is an operational aggregation score rather than a calibrated posterior probability.

$$y^{*} = \text{malicious } \textit{if } c^{*} > 0.5; \text{ benign } \textit{if } c^{*} < 0.5; \text{ manual review } \textit{otherwise} \quad (3)$$

The threshold τ governs only whether ACM requests additional analysis; the final verdict introduces no separately tuned threshold. Because c* lies on a common malicious-class scale, Eq. (3) compares it with the neutral midpoint 0.5: values above 0.5 indicate greater RFS-weighted support for the malicious class, and values below 0.5 greater support for the benign class. In practice, specimens within a narrow band $|c^{*} - 0.5| < \delta$ are routed to manual review. No additional manual-review margin was applied ($\delta = 0$), and exact ties were handled according to Eq. (3).

Supplementary Figure S3 shows how these triggers route a specimen through evidence exchange and, when needed, escalated arbitration.

If two iterations do not resolve the verdict, the Coordinator switches to escalated arbitration (Supplementary Figure S3). The threat-intelligence agent relaxes its similarity criteria and searches full

incident reports for additional evidence, and the Coordinator then requests a final reassessment using the complete evidence from all three agents. This step reuses the accumulated-evidence structure of the QACP context component and requires no protocol extension. If disagreement persists, Eq. (3) is applied and the unresolved status is recorded in the report.

The threshold τ was swept from 0.55 to 0.75 in steps of 0.05, and $\tau = 0.65$ was selected as the operating point. The sweep used the same 1,000 specimens as the performance evaluation (Section 6.3).

3.6 Offline RAG-Based Threat Intelligence

The threat-intelligence agent queries an offline RAG repository stored on the analysis server and attaches the retrieved sources to its output, so that analysts can trace each family or technique attribution to a reference.

The repository contains MITRE ATT&CK Enterprise and ATT&CK for ICS descriptions [31], malware-family profiles, historical incident reports and CVE summaries. To prevent leakage, the 1,000 evaluation specimens, any analyses derived from them and the evaluation reference labels were excluded from the repository.

4. Experimental Setup and Results

4.1 Hardware and Software Environment

Experiments used a single NVIDIA GB10 Grace Blackwell system with a 20-core Arm CPU and 128 GB of LPDDR5x unified memory. ARM64 PyTorch and vLLM served four 4-bit-quantized models, all resident in unified memory; model calls were sequential. The ACM threshold was $\tau = 0.65$, with no more than two re-verification iterations. Final classification used Eq. (3).

4.2 Dataset

Table 1 lists the 1,000 EMBER 2018-based specimens: 200 each of ransomware, backdoor/RAT, trojan and worm, plus 200 benign specimens.

Table 1
Dataset Composition

Category	No. of Specimens	Ratio (%)
Ransomware	200	20.00
Backdoor/RAT	200	20.00
Trojan	200	20.00
Worm	200	20.00
Benign	200	20.00
Total	1,000	100.00

EMBER 2018 [32] provides static PE features with malicious/benign labels. Specimens were drawn from its evaluation partition, deduplicated and assigned to the four malware categories according to their family labels. Agents received structured feature summaries and the accumulated QACP evidence rather than raw PE binaries, so the behavior agent inferred behavior from static evidence. Specimen identifiers and category-mapping rules are available from the corresponding author (see Data availability). Ground-truth malicious/benign labels and AVClass-derived family annotations were used solely for evaluation and were excluded from the feature summaries provided to the agents.

The dataset represents Windows PE threats that may reach IT-side hosts in critical infrastructure.

4.3 Evaluation Metrics

Equations (4)–(9) evaluate detection performance and operational suitability. TP, FP, TN, and FN are the numbers of true positives, false positives, true negatives, and false negatives at the specimen level. Thus, TP + FN is the number of malicious specimens and FP + TN is the number of benign specimens. Precision P is TP/(TP + FP), recall R equals the detection rate DR, N is the number of analyzed specimens in the timing population, and t_i is the elapsed time from input to final report for specimen i.

$$DR = TP / (TP + FN) \quad (4)$$

$$FPR = FP / (FP + TN) \quad (5)$$

$$P = TP / (TP + FP),\ R = DR,\ F_1 = 2PR / (P + R) \quad (6)$$

$$FCA = N_{correct-family} / N_{detected-malicious} \quad (7)$$

$$ATA = N_{correct-ATT\&CK} / N_{total-technique} \quad (8)$$

$$MTTR = (1/N) \sum_{i=1}^{N} t_i \quad (9)$$

DR is malicious-specimen recall; FPR is the proportion of benign specimens labeled malicious; and F1 is the harmonic mean of precision and recall. FCA is computed among detected malicious specimens, excluding missed malware. ATA measures the proportion of reference ATT&CK technique labels matched by the model output, not specimen-level classification accuracy. MTTR is the mean end-to-end analysis time over the specified population. Ninety-five-percent confidence intervals (CIs) for the proportion metrics (DR, FPR, precision, FCA) are Wilson score intervals, and the F1 interval is obtained by resampling the reported confusion-matrix counts under a binomial model (20,000 iterations).

ATT&CK reference labels. Because EMBER 2018 does not provide ATT&CK technique labels, the authors mapped each specimen's static indicators and family information to ATT&CK definitions and cross-checked the resulting reference labels independently of MACF outputs. ATA therefore measures agreement with this study-specific reference set. Because the reference labels and the model share the same static indicators as input, ATA should be read as consistency with a static-evidence-based reference rather than agreement with independent ground truth.

4.4 Overall Detection Performance

Table 2 compares MACF with four single-LLM baselines on the same 1,000 specimens; each baseline received the identical structured PE features and returned the same verdict–confidence–family–technique schema as MACF, but used one model for all judgments without QACP exchange, ACM re-verification, or access to the offline threat-intelligence repository. MACF yielded TP = 768, FP = 17, TN = 183 and FN = 32, giving DR = 96.0% (95% CI 94.4–97.2), FPR = 8.5% (5.4–13.2), precision = 97.8% (96.6–98.6) and F1 = 96.9% (96.0–97.7). It assigned the correct family to 638 of the 768 detected malicious specimens (FCA = 83.1%, 80.3–85.6) and matched 75.4% of the reference ATT&CK labels. The strongest baseline, EXAONE-Deep 32B, reached DR = 86.8% (84.2–88.9), F1 = 91.4%, FCA = 72.5% and ATA = 66.7%; MACF improved on it by 9.2 percentage points in DR, 10.6 points in FCA and 8.7 points in ATA.

Table 2
Detection and Interpretation Performance: Single-Model LLMs versus MACF

System (Model Configuration)	DR (%)	FPR (%)	Precision (%)	F1 (%)	FCA (%)	ATA (%)
Single · Llama-3.3 70B	79.5	17.5	94.8	86.5	62.0	54.7
Single · Qwen2.5-Coder 32B	81.8	16.0	95.3	88.0	65.6	58.5
Single · Qwen3 32B	84.5	14.0	96.0	89.9	69.1	63.1
Single · EXAONE-Deep 32B	86.8	12.0	96.7	91.4	72.5	66.7
Integrated MACF	**96.0**	**8.5**	**97.8**	**96.9**	**83.1**	**75.4**

MACF also had the lowest FPR (8.5%, versus 12.0–17.5% for the baselines), so the gain in detection did not come at the cost of additional false alarms.

4.5 Performance by Malware Type

Table 3 breaks down performance by malware category. DR was highest for worms (97.5%), followed by ransomware (96.0%), trojans (95.5%) and backdoors/RATs (95.0%). Backdoors/RATs also had the lowest FCA (80.5%) and the longest mean analysis time (66.8 s). The mean end-to-end time across malicious specimens was 63.0 s.

Table 3
Performance by Malware Type (800 malicious specimens)

Type	DR (%)	FCA (%)	ATA (%)	MTTR (s)
Ransomware	96.0	84.4	77.5	61.2
Backdoor/RAT	95.0	80.5	74.0	66.8
Trojan	95.5	81.7	72.2	63.5
Worm	97.5	85.6	77.8	60.4
Overall (malicious)	96.0	83.1	75.4	63.0

4.6 Ablation Study

Table 4 compares six configurations: (a) full MACF; (b)–(d) MACF with the static, behavior or threat-intelligence agent removed; (e) MACF without ACM, using simple aggregation; and (f) MACF with EXAONE-Deep 32B in all four roles, which keeps the roles, QACP and ACM but removes model heterogeneity. Configurations (a)–(d) retain ACM, including its low-confidence re-verification trigger, while configuration (e) disables ACM and aggregates all agents in a single RFS-weighted pass. The ablations were scored on binary detection.

Table 4
Ablation Study Results

Configuration	TP (of 800)	FN	DR (%)	ΔDR (pp)
(a) Full MACF	768	32	96.0	—
(b) Static Agent Removed	714	86	89.3	−6.7
(c) Behavior Agent Removed	766	34	95.8	−0.2
(d) Threat Intelligence Agent Removed	758	42	94.8	−1.2
(e) ACM Removed (Simple Aggregation)	737	63	92.1	−3.9
(f) Heterogeneity Removed	709	91	88.6	−7.4

Removing the static agent caused the largest single-agent loss, lowering DR from 96.0% to 89.3% (− 6.7 points). Removing the threat-intelligence agent lowered DR to 94.8% (− 1.2 points), and removing the behavior agent lowered it to 95.8% (− 0.2 points). The 95% confidence intervals describe the uncertainty in each configuration's detection rate (static-agent removed 89.3%, 86.9–91.2; threat-intelligence removed 94.8%, 93.0–96.1; behavior removed 95.8%, 94.1–96.9; full MACF 96.0%, 94.4–97.2). Formal

paired significance testing was not performed because specimen-level paired prediction records were unavailable. Because ACM remained active in these single-agent ablations, each reduction isolates the effect of removing that agent with the rest of the pipeline intact. Static evidence accounts for the largest share of binary detection, whereas the behavior and threat-intelligence agents are designed chiefly to supply family and technique interpretation, which this binary ablation does not capture.

Removing ACM reduced the detection rate from 96.0% to 92.1% (− 3.9 points; CI 90.1–93.8).

Assigning EXAONE-Deep 32B to all four roles lowered DR to 88.6% (− 7.4 points; CI 86.2–90.6), the largest drop in the study. Because the roles, QACP and ACM were held constant, this gap reflects the combined effect of role-specific model assignment, including differences in total model capacity and inference cost (Section 6.3).

4.7 Resource Efficiency and Operational Suitability

Table 5 shows that the four 4-bit-quantized models, with about 166 billion parameters in total, occupied approximately 103.0 GB, or 80.5% of the 128-GB unified memory, and remained resident simultaneously with sequential inference. The per-agent single-pass latencies summed to 19.6 s. Including conditional re-verification and report generation, the mean end-to-end time was 63.0 s for the 800 malicious specimens and 61.7 s for all 1,000 specimens.

Table 5
Per-Agent Resource Use and Inference Latency

Agent	Applied Model	Memory (4-bit)	Latency	Simultaneous Residency
CO	EXAONE-Deep 32B	19.8 GB	4.8 s	○
Static Analysis	Qwen2.5-Coder 32B	19.5 GB	4.2 s	○
Behavior Analysis	Qwen3 32B	20.1 GB	4.5 s	○
Threat Intelligence	Llama-3.3 70B + RAG	43.6 GB	6.1 s	○
Integrated MACF	—	**103.0 GB**	**19.6 s**	○

4.8 Discussion of Results

Three findings stand out. First, coordinating role-assigned local models outperformed every single model: MACF exceeded the strongest single LLM by 9.2 points in DR and 10.6 points in FCA while also lowering FPR. Second, both the choice of model for each role and conditional re-verification matter: replacing role-specific assignment with a single model cost 7.4 points of DR, and removing ACM cost 3.9 points. Third, static evidence is the backbone of binary detection, as the 6.7-point loss without the static agent shows.

The heterogeneous configuration was chosen because the candidate models showed different strengths across roles (Supplementary Table S1). The results confirm that this assignment performs better as a whole. They do not by themselves show that heterogeneous models make independent errors; establishing that would require specimen-level analysis of cross-agent error correlation.

MACF produced 17 false positives among 200 benign specimens and 32 false negatives among 800 malicious specimens. An FPR of 8.5% is too high for fully automated blocking, so MACF is intended as a triage and analyst-support tool: it prioritizes suspicious files and presents each verdict with its evidence, while response actions remain subject to human review.

5. Illustrative ICS/OT Threat Scenarios

5.1 Scenario Design

Two qualitative scenarios illustrate how MACF would operate for a hypothetical operator of power-generation and substation facilities. Scenario 1 follows a known IT-side PE malware sample along the direct triage path; Scenario 2 follows an ambiguous, suspected ICS-targeted sample that requires re-verification and expanded offline retrieval. Neither scenario is part of the labeled evaluation in Section 4. The feature evidence shown is representative of an operational deployment and, unlike the EMBER-based evaluation, may include string- and import-level detail; the scenarios therefore illustrate the coordination workflow rather than reproduce the evaluation's feature set. All confidence and time values reported for the two scenarios are illustrative of the workflow and are not measured results.

Scenario 1 assumes a suspicious file introduced through removable media. Scenario 2 assumes a file whose control-system-related indicators leave the initial verdict uncertain.

5.2 Per-Scenario Analysis Results

5.2.1 Scenario 1: Known IT Malware Infiltration

In Scenario 1 (Supplementary Table S2), the static agent reports a malicious verdict with confidence 0.91 on the basis of cryptographic imports and ransom-note strings, and the behavior agent reports 0.88 for inferred file enumeration and encryption. Both confidences exceed $\tau = 0.65$, so ACM is not invoked. The threat-intelligence agent retrieves a matching ransomware family (similarity 0.85) and reports a malicious verdict. The Coordinator combines the agent outputs into an aggregated score of $c^* \approx 0.88$ and issues a ransomware triage report, with a cumulative processing time of about 61 s.

5.2.2 Scenario 2: Suspected ICS-Targeted Malware (Illustrative)

In Scenario 2 (Supplementary Table S3), the static agent finds structural anomalies and control-related strings, and the behavior agent infers unauthorized-command and abnormal-communication indicators.

The offline repository supplies incident and ATT&CK for ICS context.

The initial confidences of 0.52 and 0.58 fall below τ and trigger ACM. Two evidence-exchange rounds raise them only to 0.61 and 0.63, so the iteration limit is reached and escalated arbitration begins. Expanded offline retrieval identifies a related ICS incident (similarity 0.71), after which the static and behavior agents report 0.72 and 0.74. The Coordinator combines the agent outputs into an aggregated score of $c^* \approx 0.72$ and issues a malicious verdict flagged as suspected ICS targeting, with a cumulative processing time of about 118 s.

5.3 Summary of Illustrative Scenarios

The two scenarios exercise the two main paths through MACF (Supplementary Table S4). Scenario 1 takes the direct path without re-verification, whereas Scenario 2 requires additional re-verification and escalated arbitration.

6. Conclusion

6.1 Summary and Implications

This paper presented MACF, a fully local multi-agent framework that combines four role-assigned LLMs with QACP, ACM and an offline knowledge base for PE-malware triage in the IT-side environments of power, energy and other critical-infrastructure networks. On 1,000 EMBER 2018-based specimens, MACF achieved 96.0% DR, 8.5% FPR, 96.9% F1, 83.1% FCA and a 75.4% ATT&CK match rate, outperforming four single-LLM baselines and a homogeneous four-role configuration, while keeping all four models resident on one 128-GB server.

6.2 Significance of the Research

The study has three implications for air-gapped operations. First, LLM-based malware triage with family and technique interpretation is feasible without any external inference API. Second, both the assignment of models to roles and the conditions under which agents re-verify one another measurably affect detection, so coordination design deserves as much attention as model choice. Third, QACP and ACM make each verdict traceable to structured agent evidence, which supports analyst review. MACF's distinct contribution is to integrate malicious detection with family and ATT&CK interpretation in a single local analysis procedure under fully offline constraints.

6.3 Limitations and Future Work

This study has seven limitations. (1) The evaluation uses structured Windows PE features rather than dynamic execution traces, OT-native binaries, firmware or industrial traffic, and the ICS/OT scenarios are qualitative. (2) It does not include packed, anti-VM or otherwise evasive specimens. (3) It omits conventional static classifiers and inference-budget-matched comparisons; the heterogeneous and homogeneous configurations also differ in model capacity, and the ablations report binary detection only. (4) RFS component weights and repetition counts were not retained in reproducible form. (5) τ =

0.65 was selected on the evaluation specimens, and agent confidences are not probability-calibrated, so c* is an operational score rather than a posterior probability. (6) The ATT&CK reference labels are study-specific, and specimen-level error records, which would be needed to attribute false positives and negatives or to test whether heterogeneous agents make independent errors, were not collected. (7) Resource measurements cover one GB10 system and one offline-knowledge configuration.

Future work should evaluate standard malware classifiers on the same data split, compare matched inference budgets, report RFS and label-generation settings, calibrate ACM on held-out data, analyze cross-agent error correlation, measure the interpretive contribution of each agent, and test evasive malware and OT-native evidence.

Declarations

Competing interests

The authors declare no competing interests.

Author Contribution

S.J. conceived the study, designed the methodology, performed the analysis and wrote the manuscript. M.L. contributed to the analysis and validation. T.S. reviewed the manuscript and provided critical revisions. All authors reviewed the manuscript.

Acknowledgements

This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-16068069).

Data Availability

The EMBER 2018 dataset is publicly available from the Elastic EMBER repository (https://github.com/elastic/ember). The identifiers of the 1,000-specimen evaluation subset and the category-mapping rules are available from the corresponding author on reasonable request.

References

1. Xu, H., Wang, S., Yu, T., Liu, Y. & Wang, H. Large language models for cyber security: a systematic literature review. *Preprint at.* (2024). https://arxiv.org/abs/2405.04760

2. Fujii, S. & Yamagishi, R. Feasibility study for supporting static malware analysis using LLM. In European Symposium on Research in Computer Security, LNCS vol. 15264Springer, (2025).
3. Falliere, N., Murchu, L. O. & Chien, E. *W32.Stuxnet Dossier, version 1.4* (Symantec Security Response, 2011).
4. Dragos Inc. TRISIS malware: analysis of safety system targeted malware. Technical report. Dragos, (2017).
5. E-ISAC & SANS ICS. *Analysis of the cyber attack on the Ukrainian power grid* (Defense Use Case (E-ISAC, 2016).
6. Stouffer, K. et al. Guide to operational technology security. *NIST Special Publication 800 – 82 Rev.* (2023). 3 (NIST.
7. International Electrotechnical Commission. *IEC 62443: Security for Industrial Automation and Control Systems* (IEC, 2018).
8. European Parliament and Council. Directive (EU) 2022/2555 on measures for a high common level of cybersecurity across the Union (NIS2 Directive). *Official J. Eur. Union* (2022).
9. Zhang, J. et al. When LLMs meet cybersecurity: a systematic literature review. *Cybersecurity* **8** (2025).
10. Yao, Y. et al. A survey on large language model security and privacy. *High-Confidence Comput* (2024).
11. Wu, Q. et al. AutoGen: enabling next-generation LLM applications via multi-agent conversation. Preprint at (2023). https://arxiv.org/abs/2308.08155
12. Hong, S. et al. MetaGPT: meta programming for a multi-agent collaborative framework. Preprint at (2023). https://arxiv.org/abs/2308.00352
13. Li, G. et al. CAMEL: communicative agents for mind exploration of large language model society. In Advances in Neural Information Processing Systems (2023).
14. Park, J. S. et al. Generative agents: interactive simulacra of human behavior. In Proc. ACM Symposium on User Interface Software and Technology (2023).
15. Du, Y. et al. Improving factuality and reasoning in language models through multiagent debate. Preprint at (2023). https://arxiv.org/abs/2305.14325
16. Liu, Z. Multi-agent collaboration in incident response with large language models. Preprint at (2024). https://arxiv.org/abs/2412.00652
17. Raff, E. et al. Malware detection by eating a whole EXE. Preprint at (2018). https://arxiv.org/abs/1710.09435
18. Or-Meir, O. et al. Dynamic malware analysis in the modern era. *ACM Comput. Surv* 52 (2019).
19. Demetrio, L. et al. Adversarial EXEmples: a survey and experimental evaluation. *ACM Trans. Priv. Secur* **24** (2021).
20. Frantar, E. et al. GPTQ: accurate post-training quantization for generative pre-trained transformers. In International Conference on Learning Representations (2023).

21. Lin, J. et al. AWQ: activation-aware weight quantization for LLM compression and acceleration. In Proc. Machine Learning and Systems (2024).
22. Xiao, G. et al. SmoothQuant: accurate and efficient post-training quantization for large language models. In International Conference on Machine Learning (2023).
23. Dettmers, T. et al. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In Advances in Neural Information Processing Systems (2022).
24. Kwon, W. et al. Efficient memory management for large language model serving with PagedAttention. In Proc. ACM Symposium on Operating Systems Principles (2023).
25. Lewis, P. et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Advances in Neural Information Processing Systems (2020).
26. Kravchik, M. & Shabtai, A. Detecting cyber attacks in industrial control systems using convolutional neural networks. In Proc. Workshop on Cyber-Physical Systems Security and Privacy (2018).
27. LG AI Research. EXAONE Deep: reasoning enhanced language models. Preprint at (2025). https://arxiv.org/abs/2503.12524
28. Hui, B. et al. Qwen2.5-Coder technical report. Preprint at (2024). https://arxiv.org/abs/2409.12186
29. Qwen Team. Qwen3 technical report. Preprint at (2025). https://arxiv.org/abs/2505.09388
30. Llama Team. The Llama 3 herd of models. Preprint at (2024). https://arxiv.org/abs/2407.21783
31. Strom, B. E. et al. MITRE,. MITRE ATT&CK: Design and Philosophy. MITRE Technical Report MP180360 (2020).
32. Anderson, H. S. & Roth, P. EMBER: an open dataset for training static PE malware machine learning models. Preprint at (2018). https://arxiv.org/abs/1804.04637

Figures

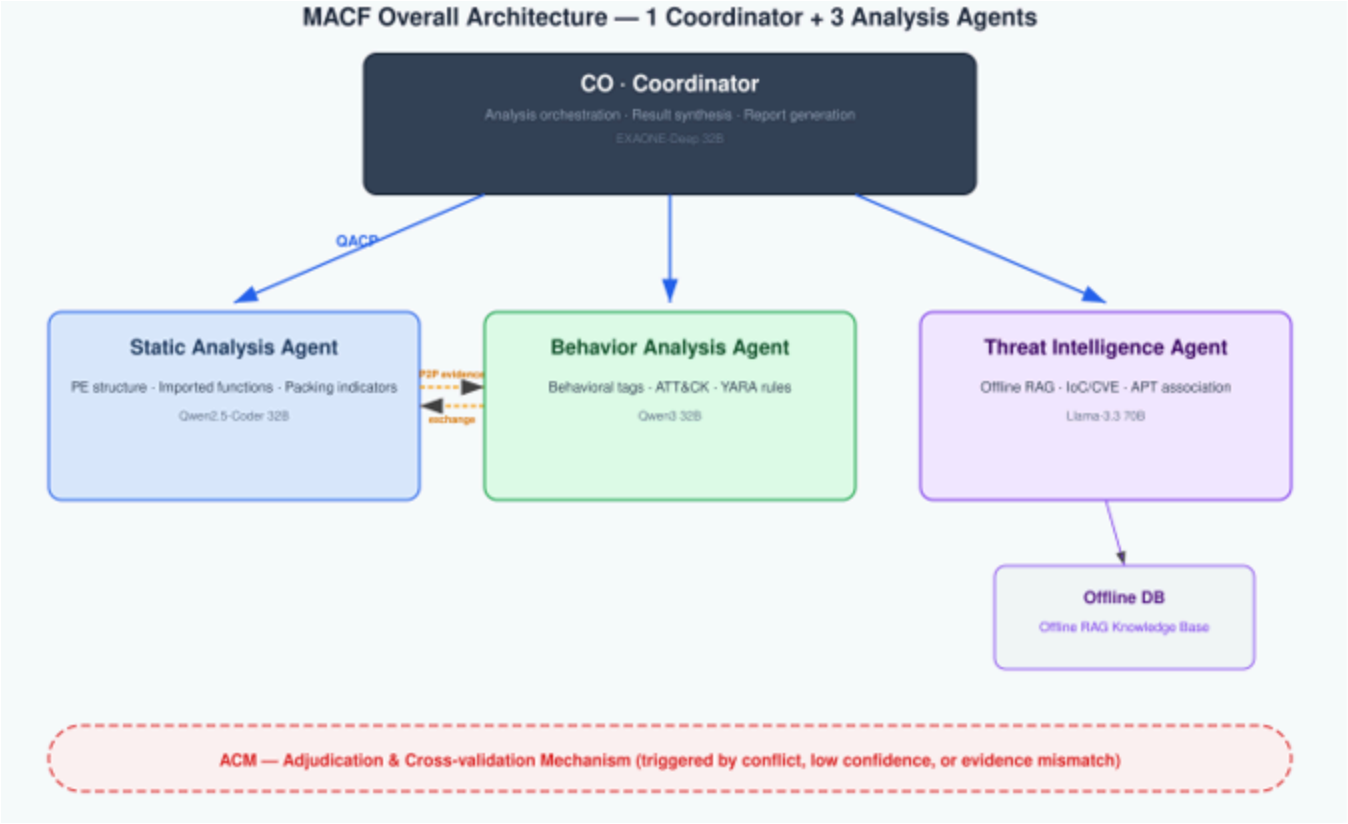

Figure 1

MACF Overall Architecture (CO + Static, Behavior, and Threat Intelligence Agents)

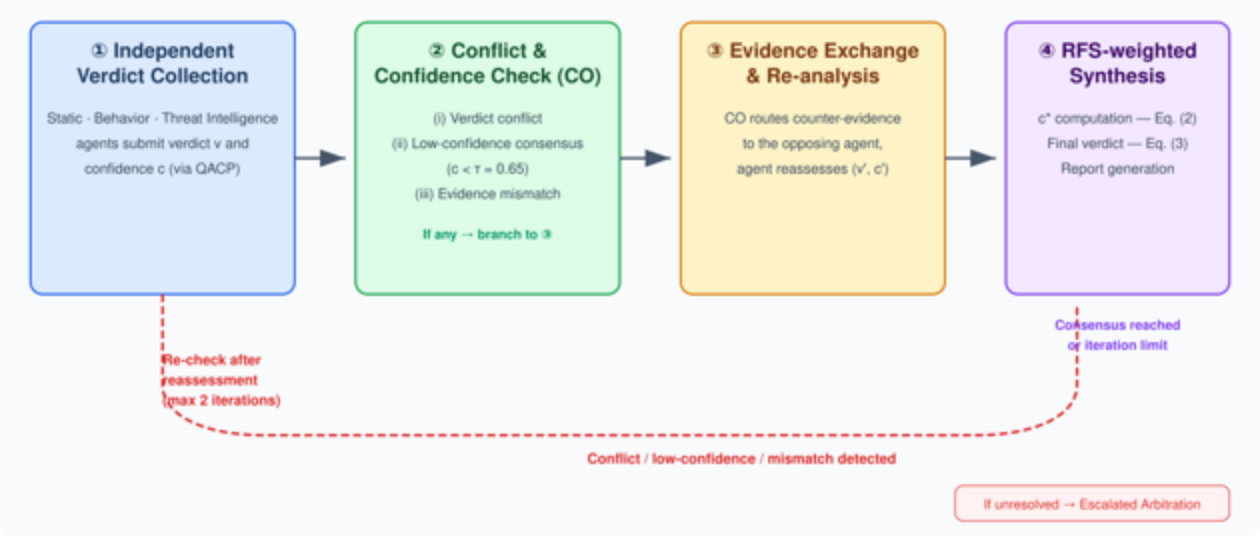

Figure 2

ACM Four-Stage Cross-Validation Procedure

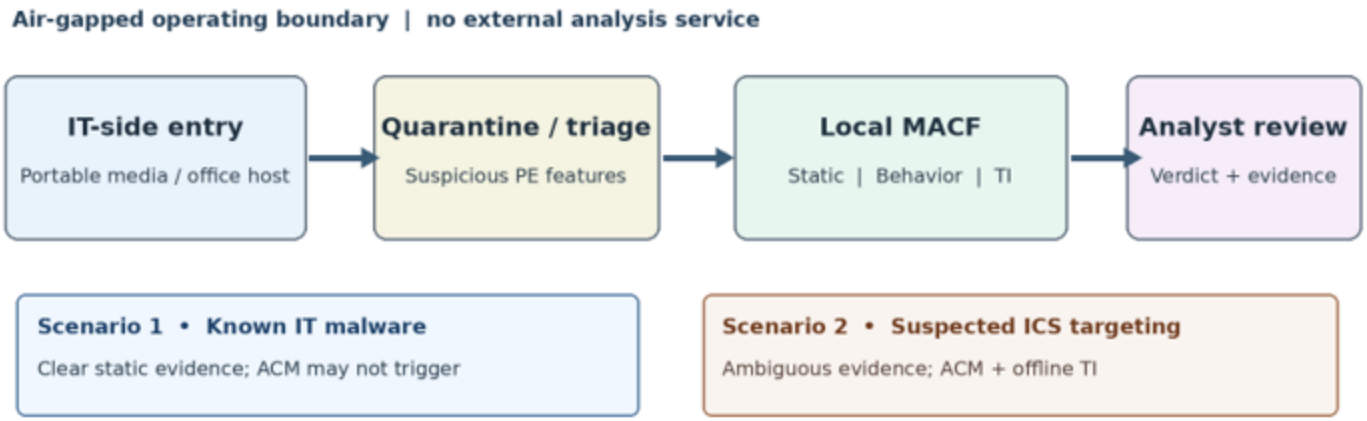

Figure 3

Conceptual Deployment and Two ICS/OT Malware Triage Scenarios