Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.088835

ARTICLE

RACE-SOC: Risk-Constrained Auditable Coordination Evaluation for LLM-Based Multi-Agent Security Operations

Yi Lv[1,*] **and Cheng Wang**[1,2,*]

[1]School of Computer Science and Technology, Tongji University, Shanghai, China
[2]Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai, China

*Corresponding Authors: Yi Lv. Email: lvyi@tongji.edu.cn; Cheng Wang. Email: cwang@tongji.edu.cn

Received: 14 July 2026; Accepted: 17 September 2026; Published: 24 November 2026

ABSTRACT: Large language model (LLM) agents are now being used in security operations center (SOC) workflows to summarise security alerts, support triage, and suggest escalation or containment measures. Existing evaluations tend to focus on answer quality or benchmark accuracy, while the operational risks of multi-agent coordination are not clearly exposed. This gap is problematic in SOC triage because extra agents may produce unsupported claims, disagreement, false consensus, or unsafe operations even when the final answer seems reasonable. This paper proposes RACE-SOC, a Risk-constrained Auditable Coordination Evaluation protocol for LLM-based multi-agent SOC triage. RACE-SOC evaluates bundled role-conditioned configurations over fixed source groups by scoring the trade-off among triage utility, coordination risk, and token cost. The corrected deterministic scorer combines utility terms with a K-invariant primary risk score based on bounded unsafe-output incidence and unsupported-claim rate; coordination-only disagreement and false-consensus terms are reported separately. Across 45 model–dataset–seed cells, the corrected operational selections are K = 1/5/10 in 13/14/18 cells, including seven K = 5/K = 10 co-best cells resolved by a preregistered cost-first tie-break. Each sampling seed realizes n = 28 source groups (6/6/4/6/6 across the five datasets), giving 84 seed-indexed observations but only 54 distinct alert-set signatures with overlap; these are not statistically independent population samples. We instantiate the protocol on five public cybersecurity datasets and compare Qwen3.5-9B local inference, GLM-5.2, and DeepSeek-v4-pro. We also add a matched same-temperature repeated-inference control at K = 5 and K = 10 to separate repeated sampling from the role-conditioned pipeline. The control obtains a higher mean net score in five of six model–K strata, while the role-conditioned pipeline is higher for GLM-5.2 at K = 10. The direction of all six comparisons is retained after unique-signature weighting and after excluding plurality ties. These mixed results do not support a general benefit from role specialization under the tested design. Instead, they show why agent count, prompt view, aggregation, and risk must be audited together. The contribution of RACE-SOC is a reproducible evaluation instrument for exposing these configuration boundaries in SOC triage rather than assuming that more roles are inherently better.

KEYWORDS: Coordination risk; alert triage; false consensus; offline replay; security evaluation

1 Introduction

Internet-facing organizations now use language-model agents to triage security events, summarize logs, and recommend escalation steps. In this setting, the output is not a passive label. It may influence analyst review, containment, isolation, or the decision to leave an alert unhandled. A useful evaluation protocol therefore has to ask more than whether an answer looks plausible. It must also record whether the answer

Copyright © 2027 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

is grounded in visible evidence, whether the recommended action is compatible with the alert context, and whether adding agents introduces coordination failures such as disagreement or false consensus.

Recently, security research has explored applications of large language models (LLMs) in threat detection, log analysis, and autonomous security operations center (SOC) workflows [1–6]. These studies show that language models can enter security analysis pipelines. The next engineering question is how to control the number of agents when evidence gain, disagreement, false consensus, latency, token cost, and unsafe action risk move together. Multi-agent systems can improve division of labor in software-engineering settings [7], but SOC triage requires a stronger audit trail because model output can affect containment, escalation, or analyst workload.

Risk-constrained Auditable Coordination Evaluation for SOC (RACE-SOC) addresses this question through a reproducible evaluation instrument. RACE-SOC defines multi-agent SOC triage as a risk-constrained decision process in which a useful output retains critical alerts, suppresses low-risk noise safely, preserves action consistency, and stays supported by visible evidence. In this paper, fixed source context means that every configuration starts from the same pre-constructed SOC triage group, with no retrieval, tool calls, memory updates, or additional investigation during the run. The actual prompt view can still vary with the assigned role and its evidence budget. We therefore evaluate bundled role-conditioned configurations, not the isolated causal effect of agent count. We evaluate K values of 1, 5, and 10 for three calibrated model routes, including Qwen3.5-9B local inference through a Windows Subsystem for Linux (WSL)/Ollama endpoint, GLM-5.2 through the BigModel endpoint, and DeepSeek-v4-pro through the DeepSeek endpoint.

The deterministic scorer is released as code in the accompanying artifact. It reads recorded experiment outputs and applies a fixed rule set without invoking an LLM judge. Every reported score can be recomputed from the JSONL outputs, visible evidence identifiers, ground-truth labels, expected actions, agent votes, and scorer rules. This design makes the rule set auditable and replaceable. The present study uses metric-based scoring as the main evidence source because a human-rating study would require a separate inter-rater agreement protocol.

The calibrated matrix uses three seeds, five datasets, three model routes, and three agent-count settings. The goal is to show how scale-dependent failure terms can be surfaced and audited when agent count changes, and whether recorded agent-count trajectories contain enough signal to motivate risk-adaptive scheduling.

The three purposes of this study are as follows:

RQ1. Quantify how the bundled configuration indexed by K—its role set, role-conditioned prompt views, generation count, and deterministic aggregation inputs—changes the joint frontier of decision quality, unsupported claims, unsafe actions, coordination diagnostics, and token cost in SOC alert analysis.

RQ2. Determine whether one fixed bundled configuration is suitable across datasets, model routes, and sampling seeds, or whether different alert settings need different risk-cost trade-offs.

RQ3. Evaluate whether offline replay over fixed agent-count traces can recover a measurable fraction of the per-stratum oracle gap and expose constraints for future risk-adaptive scheduling.

The paper makes four contributions. First, it defines RACE-SOC, a risk-constrained evaluation framework for multi-agent SOC triage in which K indexes an auditable bundle of role composition, evidence allocation, generation count, and aggregation inputs rather than an isolated causal treatment. Second, it operationalizes eight code-anchored metrics that combine positive triage utility with coordination-failure penalties, including unsupported claims, unsafe-action violation density, disagreement, and false consensus, and it specifies the scorer in full so that the rule set itself can be audited and reused. Third, it reports a calibrated strong-model pilot over five public datasets using Qwen3.5-9B, GLM-5.2, and DeepSeek-v4-pro, together with a three-seed best-K sampling-sensitivity audit and a lambda sensitivity audit. Fourth, it adds

a paired same-temperature repeated-inference control at K = 5 and K = 10, together with unique-signature, aggregation-tie, and evidence-exposure sensitivity analyses. The present paper does not claim to build an online adaptive SOC scheduler, but it shows which failure, cost, and sensitivity signals such a scheduler would have to control.

2 Related Work

We position RACE-SOC against four adjacent research lines and tie each citation to a specific design decision, rather than cataloguing the field. The distinctions we draw are intended to clarify what RACE-SOC is not. It is not an autonomous SOC agent, an alert-compression system, a provenance-based detector, or a benchmark that ranks models inferentially.

The closest line is LLM-driven SOC automation. ChatSOC places LLM agents inside an SOC workflow and is the nearest analogue to the present task [1]. Guided-response systems and cognitive-SOC surveys describe how language models can be embedded in operational response and how AI is entering the SOC more broadly [5,6]. RACE-SOC differs from this line in its object of evaluation. These systems are designed to perform SOC work. Their evaluation typically asks whether the assisted workflow is useful. RACE-SOC instead uses the number of agents as a fixed-K stress variable and asks where added roles stop improving evidence-supported output and begin adding coordination failure. This design decision motivates our explicit disagreement and false-consensus penalties. The aim is not to replace adaptive SOC workflows, but to produce recorded trajectories from which risk-adaptive scheduling constraints can be audited.

A second line is LLM-based log and threat analysis. LogPrecis applies language models to malicious-log analysis [2], and survey and systematic-review studies have examined broader applications of LLMs in cyber threat detection and security engineering [3,4]. Knowledge-graph approaches construct threat intelligence and explainable detection from model outputs [8,9]. This line usually aims to optimise a detection or interpretation object. It informs the output contract used here, especially the requirement that a triage output list supported and unsupported claims with explicit evidence references. It does not directly audit coordination cost because it does not evaluate how risk, cost, and evidence support change when fixed coordination scale varies over the same controlled triage unit.

A third line is alert fatigue, prioritisation, and aggregation. Human-AI teaming, machine-learning and reinforcement-learning prioritization, SIEM operations, and performance metrics for SOCs all address the volume problem that motivates triage [10–16]. AECID addresses domain-independent alert aggregation and is the relevant traditional compression reference [17]. We include AECID to mark a construction difference. Aggregation-based approaches reduce alert volume by grouping or compressing alerts before analyst attention is allocated. RACE-SOC does not aggregate alerts. Its unit of evaluation is a triage unit tau = (a, E, A) in which the alert or alert group, the visible evidence, and the candidate actions are fixed in advance, and the variable under test is how a multi-agent system reasons over that fixed unit. RACE-SOC evaluates the safety and evidence support of reasoning after triage-unit construction. It does not evaluate pre-triage alert compression.

A fourth line is provenance-based investigation and log anomaly detection. Alert correlation, automated provenance triage, multi-layer log analysis, runtime provenance detection, root-cause analysis, and deep log anomaly detection provide foundations for SOC automation [18–25]. These methods optimize detection fidelity, causal reconstruction, or alert reduction over system telemetry. RACE-SOC instead treats the multi-agent reasoning process itself as the object under test and scores its outputs with a transparent rule set. It does not propose a new detector. Public datasets such as the AIT Alert Data Set (AIT-ADS) [26], the Los Alamos National Laboratory cyber-security event dataset (LANL) [27], the Spanish Research National Council HTTP dataset (CSIC2010) [28], the University of New South Wales Network Benchmark 2015 dataset

(UNSW-NB15) [29], and the Canadian Institute for Cybersecurity Intrusion Detection System 2017 dataset (CICIDS2017) [30] are used here as triage-unit sources rather than as training corpora for a new classifier.

RACE-SOC adds an additional assessment dimension across these four research lines. It makes fixed agent-count trajectories and their failure terms jointly auditable under a deterministic, openly specified scorer. Table 1 sharpens this positioning by recording whether each research line directly evaluates agent count, coordination risk, operational action, and validation basis. RACE-SOC uses deterministic rules so that other researchers can reproduce, inspect, and modify the scoring procedure.

Table 1: Positioning of RACE-SOC relative to adjacent research lines.

Research Line	K	Coord. Risk	Risk Metric	SOC Action	Validation Basis
SOC LLM agents [1,5,6]	No	Partial	Partial	Yes	Human or benchmark evaluation
LLM log and threat analysis [2–4,8,9]	No	No	Partial	Partial	Benchmark evaluation
Alert prioritization and aggregation [10–17]	No	No	Yes	Partial	Benchmark or human evaluation
Provenance and log-anomaly investigation [18–25]	No	No	Yes	Partial	Benchmark evaluation
OURS: RACE-SOC	Yes	Yes	Yes	Yes	Deterministic rules, external validation remains open

Note: RACE-SOC is distinguished by auditing fixed-K coordination risk, cost, and evidence support under a deterministic scorer.

Multi-agent coordination outside security, exemplified by communicative software-development agents [7] and general LLM-agent collaboration frameworks [31–33], gives a broader reason to expect role specialization to help or harm depending on coordination cost. We cite this line only to ground the design decision to make disagreement and false consensus explicit penalties rather than implicit assumptions. Concerns about the decision-loop behavior of agentic AI in security further motivate treating action-bearing output as risk-bearing [34].

Sparse communication and multi-agent debate-topology methods [35–37] provide useful context for future routing extensions, but routing is not treated as a main empirical claim in this paper. The present paper keeps the main result focused on fixed-source role voting and deterministic risk scoring. Evidence routing remains a natural extension, but it must be rerun under the same model, seed, token-budget, and retry rules before it can be merged with the calibrated main matrix.

2.1 Component Security and Threat Boundary

RACE-SOC evaluates coordination quality; it is not a complete security assurance argument for every component in an LLM-based SOC. Foundation models may contain clean-label or instruction-triggered backdoors, and a customized system prompt may itself become a malicious control surface [38,39]. These risks are upstream of the fixed-K comparison and cannot be inferred from a benign-trace score. We therefore separate the measured coordination boundary from the wider component threat model in Table 2.

Table 2: Component-security matrix: threat model and residual boundaries.

Component	Representative Attack or Failure	Measured Here	Present Control	Residual Boundary
Foundation model	Training-data backdoor, hidden trigger, or provider-side model substitution	No	Exact route identifier and raw-response provenance	Model weights and provider internals remain outside the audit
System/role instruction	Backdoored instruction, prompt injection, or role-specific steering	Partly	Frozen prompt templates, fixed role sets, common-view control, prompt hashes	No adversarial instruction campaign or metacognitive-probing defence is evaluated
Parallel role votes	Unsupported claims, disagreement, or false consensus	Yes	Stored votes, micro-UCR, and K = 5/10 coordination diagnostics	Metrics are operational rules, not expert semantic adjudication
Deterministic aggregator	Tie manipulation, malformed vote, or implementation drift	Yes	Schema validation, canonical signatures, deterministic tie-break, replay tests	A compromised runtime could still alter code or inputs outside the hashed package
Parser and retry layer	Truncation, empty content, schema failure, or retry-selection bias	Yes	Attempt-level logs, bounded retries, fail-closed parsing, terminal-output accounting	Historical lower-level transport retries are not fully recoverable
Scorer and hidden keys	Label leakage, incorrect expected action, or metric gaming	Partly	Model-hidden labels, versioned scorer, fixtures, sensitivity analysis	Labels and rules are not an external analyst-agreement study
Software/ provider supply chain	Dependency, endpoint, credential, or service compromise	No	Version and hash manifests, local key isolation, route-specific audit logs	No signed model attestation or end-to-end provider verification is claimed

The table is intentionally conservative. A PASS under the RACE-SOC scorer means that a recorded output satisfied the declared operational rules; it does not certify that the underlying model, instructions, dependencies, or provider are free of backdoors or supply-chain compromise.

3 Evaluation Protocol

A source triage group is $G_{d,s,i} = \{x_j\}$, where d indexes the dataset, s the sampling seed, i the sampled group, and each x_j contains model-visible alert fields. Hidden truth labels and expected actions, denoted by $H(G)$, are available only to the deterministic scorer. The evaluated configuration set is $\mathcal{K} = \{1, 5, 10\}$. For each $K \in \mathcal{K}$, $\mathcal{R}_K$ is the fixed role set and $|\mathcal{R}_K| = K$. The implementation constructs the prompt view for role $r \in \mathcal{R}_K$ as

$$V_{K,r}(G) = B_{b_K}(\Pi_{K,i,r}(F_r(G))), \qquad b_K = \max(256, \lfloor T/K \rfloor), \tag{1}$$

where F_r is the role-specific field filter, $\Pi_{K,i,r}$ is the deterministic evidence ordering used by the runner, B_{b_K} truncates the view to the per-role evidence budget, and $T = 3000$ is the historical total evidence-selection budget. Thus the source group is held fixed across K within a model–dataset–seed cell, but the actual prompt view is not byte-identical. No configuration retrieves additional context, calls tools, communicates between roles, or updates evidence during the run.

Let p_r denote the role prompt and M_θ the frozen model-route configuration. Each role independently produces a structured vote $z_{K,r} = M_\theta(p_r, V_{K,r}(G))$. The final output is

$$y_K = \mathcal{A}(\{z_{K,r} : r \in \mathcal{R}_K\}), \tag{2}$$

where $\mathcal{A}$ is a deterministic rule aggregator, not an LLM call. We therefore define the measured object as the bundled configuration $C_K = (\mathcal{R}_K, \{V_{K,r}\}, M_\theta, \mathcal{A})$. Changing K changes role composition, prompt views, generation count, and aggregation inputs together; it does not identify the isolated causal effect of agent count. The evaluation must reward useful evidence and penalize action risk because the output is action-bearing and a persuasive but unsupported recommendation can be more harmful than a conservative escalation.

The notation maps directly to the released implementation. F_r is implemented by `ROLE_FIELDS`; $\Pi_{K,i,r}$ and B_{b_K} by `role_visible_items` and the per-role budget in `run_one_vote`; $z_{K,r}$ by the independently scheduled role requests in `run_llm_configuration`; $\mathcal{A}$ by `aggregate_votes`; and Φ by the versioned primary scorer.

RACE-SOC uses a risk-constrained score. The quality term combines critical-alert retention (CAR), false-positive reduction (FPRR), action consistency (ACR), and evidence support (ESR). The primary risk term combines bounded unsafe-output incidence over all scored outputs (UVD_{all}) and the micro-averaged unsupported-claim rate (UCR_{micro}). Agent disagreement and false consensus require at least two eligible non-manager votes, so they are reported only as K = 5/K = 10 coordination diagnostics and never enter the all-K ranking. Let M denote the base model, D denote the dataset and task structure, and λ denote the risk penalty weight. The deterministic scorer $\Phi(y_K, \{z_{K,r}\}, H(G))$ returns the component values from which the score is defined as

$$S(K, M, D, s) = Q(K, M, D, s) - \lambda R_{primary}(K, M, D, s). \tag{3}$$

In Eq. (3), larger S is better. The main tables use $\lambda = 1$, and Section 6 sweeps $\lambda \in \{0.25, 0.5, 1, 2, 4\}$. We use $Q = (CAR + FPRR + ACR + ESR)/4$, $R_{primary} = (UVD_{all} + UCR_{micro})/2$, and $K^*(M, D, s) = \arg\max_K S(K, M, D, s)$ over $K \in \mathcal{K}$, with co-best values retained within 10^{-12} and an operational cost-first tie-break. This definition places every ranked configuration on the same bounded risk scale. Legacy, capped, saturating, macro-UCR, and all-K outcome-risk variants are retained as prespecified sensitivity analyses rather than selected after viewing the corrected results.

K is the total number of stored role votes for a triage group. K = 1 uses one generalist. K = 5 and K = 10 include one role named manager and four or nine other roles, respectively. All roles, including the manager-role, respond independently and in parallel. The manager-role does not read the other votes and does not perform a second LLM synthesis. After all votes are stored, a deterministic rule aggregator produces the final structured output. K therefore changes role composition, per-role evidence allocation, total generation count, and aggregation inputs at the same time.

Each JavaScript Object Notation (JSON) output reports risk level, recommended action, human-approval flag, evidence references, supported claims, unsupported claims, and per-agent votes. Evidence references must match visible alert identifiers. A non-matching reference is treated as unsupported by the scorer. Claim attribution remains at the stored role-vote level. At K = 5 and K = 10, claims made by the independently generated manager-role vote enter UCR_{micro} exactly like claims from the other roles. The deterministic aggregator pools these recorded lists into the final JSON but creates no new model-generated claim; there is therefore no unattributed post-vote manager-synthesis claim channel in this implementation.

In the figures and tables, R denotes $R_{primary}$ unless an alternative risk definition is explicitly named.

Fig. 1 summarizes the RACE-SOC evaluation pipeline. The same SOC triage source group is replayed through the measured bundled configurations, role-conditioned views are generated without inter-agent communication, and the deterministic aggregator and scorer decompose recorded JSONL outputs into utility and risk terms.

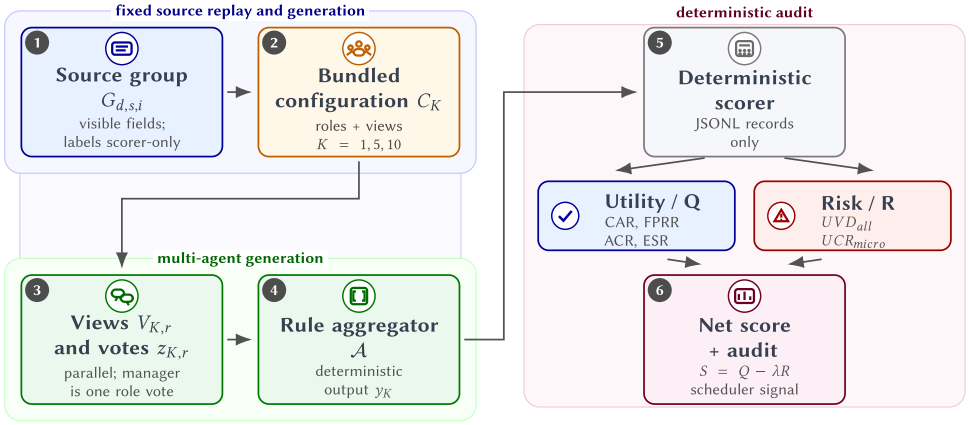


Figure 1: Formal RACE-SOC evaluation pipeline. Each configuration starts from the same source group $G_{d,s,i}$, while role filtering, ordering, and the per-role budget produce $V_{K,r}$. All K roles vote independently; the role named manager at K = 5/10 is one such vote and does not read the others. The deterministic rule aggregator $\mathcal{A}$ produces y_K, and the scorer combines y_K, stored role votes, and scorer-only labels into utility, primary risk, net scores, and audit tables.

The action-routing logic distinguishes outputs that proceed to human triage from outputs that do not. Outputs routed to human triage are the locus of the safety constraints, because they correspond to decisions with operational consequence. The action-consistency metric branches on whether a triage unit contains non-low-risk alerts. When such alerts are present, consistency is assessed against expected nonbenign handling. When a unit contains only low-risk alerts, consistency is assessed over all output actions for that unit. This branching is a deliberate design decision to avoid penalizing a system for the absence of nonbenign actions when there is no nonbenign alert to handle, and its effect on cross-dataset comparability is recorded as a construct-validity limitation.

The scorer is deterministic and artifact-based. It reads JSONL outputs, visible evidence references, alert identifiers, ground-truth risk labels, expected actions, and agent votes, and it applies a fixed rule set. It does

not use an LLM judge. Reproducibility and inspectability are therefore the basis for the audit. The scorer code and metric anchors are included in the public artifact so that the reported quantities can be recomputed and the rules can be inspected. We do not use human judgment as a main result because subjective expert ratings would require a separately designed inter-rater agreement study. The present study therefore focuses on reproducible metric-based evidence.

3.1 Routing Extensions and Scope Boundary

Evidence-centric routing is adjacent to RACE-SOC but is not part of the calibrated main matrix reported in this revision. A routing operator could select agent outputs that add non-redundant evidence, increase evidence-field diversity, and avoid low-diversity consensus before verifier recombination. Such a design would test whether the composition of messages sent to a verifier matters in addition to the number of agents.

We keep this idea outside the present empirical claim for a reproducible reason. The calibrated matrix fixes Qwen3.5-9B, GLM-5.2, and DeepSeek-v4-pro under three seeds, a 4096-token output cap, and a parse-failure retry rule. Routing results produced under earlier model or token settings would mix experimental conditions. In this version, RACE-SOC therefore evaluates fixed-source parallel role voting. Routing, planner-executor-verifier workflows, and dynamic tool-query loops are left as future extensions that require their own calibrated matrices.

4 Scorer Specification

This section specifies the scorer in full, because the scorer is the object of study and its auditability rests on the transparency of its rules. Every metric is computed from recorded experiment outputs, and no model call is involved in the audit. Algorithm 1 summarizes the deterministic evaluation flow, and Table 3 gives the full name, range, direction, and operational boundary of each metric.

Algorithm 1: RACE-SOC deterministic scorer

Input Source groups $G_{d,s,i}$, hidden keys $H(G)$, role votes $z_{K,r}$, aggregated outputs y_K, $\mathcal{K} = \{1, 5, 10\}$, and λ.

Output Metric components, Q, $R_{primary}$, S, co-best set, operational K^*, and audit logs.

for each m, d, s, K **do**
  parse y_K and every stored $z_{K,r}$; fail closed on schema or group mismatch
  ground evidence IDs against G and evaluate actions against scorer-only $H(G)$
  compute $CAR, FPRR, ACR, ESR, UVD_{all}, UCR_{micro}$ over the cell
  $Q \leftarrow (CAR + FPRR + ACR + ESR)/4$
  $R_{primary} \leftarrow (UVD_{all} + UCR_{micro})/2$
  $S_{m,d,s,K} \leftarrow Q - \lambda R_{primary}$
end for
for K = 5/10, compute ADR_{coord} and FCR_{coord} as diagnostics
retain the co-best set within 10^{-12}; choose operational $K^*_{m,d,s}$ by the cost-first tie-break
return metrics, diagnostics, Q, $R_{primary}$, S, K^*, parse status, and logs

Table 3: RACE-SOC metric glossary and operational boundaries.

Symbol	Full Name	Range/Direction	Operational Boundary
CAR	Critical-alert retention	[0, 1], higher better	Retained high-truth alerts divided by all high-truth alerts.
FPRR	False-positive reduction	[0, 1], higher better	Low-truth alerts not retained for human triage divided by all low-truth alerts.
ACR	Action consistency	[0, 1], higher better	Expected-action compatibility within the applicable nonbenign branch; if no such alert exists, the scorer uses its documented all-output fallback.
ESR	Evidence support	[0, 1], higher better	Syntactic grounding of cited alert identifiers in the visible input; it is not expert semantic validation.
UVD_{all}	Bounded unsafe-output incidence	[0, 1], lower better	Outputs with one or more unsafe-action events divided by all scored outputs; one output contributes at most one.
UCR_{micro}	Micro unsupported-claim rate	[0, 1], lower better	Unsupported claims divided by all explicit claims across stored role votes, including the independent manager-role vote at K = 5/10.
ADR_{coord}	Coordination disagreement rate	[0, 1], diagnostic	K = 5/10 outputs in which eligible non-manager votes disagree on risk or action; K = 1 is NA.
FCR_{coord}	Coordination false-consensus rate	[0, 1], diagnostic	K = 5/10 all-low groups for which at least two thirds of eligible non-manager votes are medium/high; K = 1 is NA.

The quality side contains four bounded metrics. Critical-alert retention (CAR) measures whether high-risk alerts remain available for human triage. False-positive reduction (FPRR) measures whether low-risk alerts are safely suppressed. Action consistency (ACR) measures whether the recommended action is compatible with the expected handling action. Its denominator and fallback depend on the applicable scorer branch, so ACR is interpreted within a dataset/branch composition rather than as a direct cross-dataset model ranking. Evidence support (ESR) measures whether the output cites visible alert identifiers. ESR is a syntactic grounding check rather than a semantic expert judgment.

The primary risk side contains two all-K metrics. UVD_{all} counts whether each scored output has at least one unsafe event, including an unsafe suppression or destructive action, and divides by all scored outputs. The aligned numerator and denominator make it bounded and comparable across K. UCR_{micro} measures the share of explicit claims that are not grounded by the rule set across all stored role votes, including the independently generated manager-role vote at K = 5 and K = 10. ADR_{coord} and FCR_{coord} require multiple eligible non-manager votes. They are therefore NA at K = 1 and are reported only as coordination diagnostics for K = 5 and K = 10.

The composite scores are directly derived from these indicators. The quality score is the unweighted mean of CAR, FPRR, ACR, and ESR. The primary failure score is the unweighted mean of UVD_{all} and UCR_{micro}. The net score is quality_score − λ· failure_score. The main tables use lambda = 1, and Section 6 reports the sensitivity analysis for $\{0.25, 0.5, 1, 2, 4\}$. The corrected scorer also reports the legacy definition, clipped and saturating legacy-UVD variants, macro-UCR, and a K-invariant final-risk outcome variant. These alternatives are sensitivity checks, not routes for choosing a preferred result after inspection.

This specification is one implementation of SOC triage quality for the present study. Another evidence-grounding rule or weighting scheme could change the reported ordering. The results are therefore shown as outputs of the RACE-SOC scorer rather than as inherent attributes of the models.

5 Experimental Setup

5.1 Dataset Sources and Triage-Unit Construction

The five public cybersecurity datasets used in the pilot cover complementary detection surfaces. Multi-step alert sequences are based on the AIT Alert Data Set (AIT-ADS) [26]. Enterprise authentication and host events come from the Los Alamos National Laboratory cyber-security event dataset (LANL) [27]. Web-request anomalies are derived from the Spanish Research National Council HTTP dataset (CSIC2010) [28]. Network-flow intrusion features come from the University of New South Wales Network Benchmark 2015 dataset (UNSW-NB15) [29]. Labeled network-traffic attack scenarios are taken from the Canadian Institute for Cybersecurity Intrusion Detection System 2017 dataset (CICIDS2017) [30]. These datasets are used here exclusively as sources of SOC triage units, not as training corpora for a new classifier. The objective is not to outperform a detector on any of them, but to construct realistic units on which multi-agent triage outputs can be scored under a single rule set.

A SOC triage unit is the atomic evaluation object of RACE-SOC. Each unit packages an initial alert or alert group together with the visible evidence fields that a downstream model and every agent configuration is permitted to consult, the ground-truth risk label used by the deterministic scorer, and the expected handling action against which recommended actions are checked. Construction from raw dataset records involves selecting an alert or alert group, attaching the visible evidence fields, and recording the truth label and expected action in a hidden key that the model never observes. The triage unit is therefore deliberately distinct from alert-aggregation constructs such as those in AECID-style compression [17]. Aggregation methods compress an alert stream into fewer representative alerts, whereas a triage unit fixes a bounded decision problem with a fixed evidence budget so that agent count, rather than alert-stream volume, can be used as a coordination stress factor. This distinction is what makes the present scoring auditable at the unit level.

Table 4 summarizes how each public dataset is converted into bounded triage units and reports the realized unit count used in the pilot.

5.2 Realized Sample Sizes

Each sampling seed realizes n = 28 source groups: six for AIT-ADS, six for LANL, four for CSIC2010, six for UNSW-NB15, and six for CICIDS2017. The three seeds therefore yield 84 seed-indexed group observations. Because the sampler changes group membership and order with the seed, these are not repeated stochastic generations on one fixed set of 28 units. The stored traces contain 54 distinct alert-set signatures, with overlap across seeds. The same source groups are then reused across model routes and K settings, so multiplying cells or role votes does not increase the independent evidence base. We therefore do not assign a statistically independent sample size to this pilot. The four CSIC2010 groups per seed also make that source especially small; its results are descriptive and do not support cross-dataset inference.

Table 4: Dataset and triage-unit construction used in the CMC pilot.

Dataset	Year	Security Surface	Visible Evidence and Label Source	n
AIT-ADS	2024	Host and multi-step alert scenarios	Detector hint, host/source/component/time/log evidence, converted alert truth risk.	6
LANL	2015	Enterprise authentication and host events	User/host identifiers, event type, time span and risk hint, converted alert truth risk.	6
CSIC2010	2010	HTTP web-request anomaly triage	Method, path, parameters, request line and detector hint, anomalous/normal mapping.	4
UNSW-NB15	2015	Network-flow intrusion triage	Flow and protocol/service features with risk hint, attack/benign mapping.	6
CICIDS2017	2017	Labeled network-traffic scenarios	Flow/security context, event type and attack label, attack/benign mapping.	6

5.3 Fixed-Source-Context Protocol

The protocol fixes the source alert group for all K settings within a model–dataset–seed cell. This prevents a larger configuration from retrieving new alerts or external context. It does not make every prompt byte-identical. Each role applies its own field selection and ordering, and the fixed total evidence budget is divided across K role votes. The actual prompt view is therefore a function of the source group, K, role, and per-role budget. This distinction is important: score differences between K settings describe the complete role-conditioned pipeline and cannot be attributed to agent count alone.

The role votes are generated without communication. At K = 5 and K = 10, the role named manager is one of the parallel respondents. It sees its own role-conditioned view, not the other stored votes. A deterministic rule aggregator then combines all recorded votes into the final JSON object. The scorer separately measures output utility, unsupported claims, and coordination diagnostics from those records.

The output contract requires each system response to be a single JSON object containing a risk level, a recommended action, a human-approval flag, evidence references, an explicit list of supported claims, an explicit list of unsupported claims, and the per-agent votes. The contract is what enables the deterministic scorer. Evidence references are syntax-checked against visible alert identifiers. Supported and unsupported claim lists are counted directly, and agent votes are read for disagreement and false-consensus detection. Outputs that do not meet the contract cannot be scored under the rule set, so the contract itself is part of the operationalisation rather than a convenience.

5.4 Models and Runtime Configuration

Three model routes are compared under the same units and the same output contract. Their official model cards or provider documentation are cited to identify the evaluated routes, not to imply a general capability ranking [40–42]. Qwen3.5-9B is run locally through WSL/Ollama behind an OpenAI-compatible API interface; this describes the request schema and does not mean that an OpenAI model is evaluated. The RTX A4500 serves as the local inference device. GLM-5.2 is called through the BigModel/Zhipu endpoint. DeepSeek-v4-pro is invoked through the DeepSeek endpoint. The DeepSeek route also uses the same structured-output contract as the other routes and applies a bounded retry rule for malformed JSON responses.

The protocol has a maximum unit size of six, agent counts K of 1, 5, and 10, three seeds (11, 22, and 33), and a maximum output length of 4096 tokens. The 4096-token cap is a control and not a production recommendation. It keeps the response budget comparable while allowing larger K values to express role votes, claims, and evidence references without an immediate length bottleneck.

The frozen runner supplies one user message per role and requests one structured JSON object. The total evidence-selection budget is 3000 estimated tokens per triage group and is divided approximately as $3000/K$ across role votes, subject to a 256-token floor. Role-specific field filters, deterministic field ordering, and a seed-indexed group order are applied before truncation. Provider requests use the route-specific historical decoding configuration recorded in the artifact: Qwen is local non-thinking inference; GLM uses its frozen provider request fields; DeepSeek uses thinking mode with the recorded effort setting. The 4096 value is a maximum output length for every role vote, not a total input budget.

Each generation is parsed against the same output schema. A malformed or empty structured response is a generation failure and may be regenerated only within the frozen three-attempt ceiling; transport failures do not silently change the model, prompt, temperature, thinking mode, concurrency, or token cap. Attempt-level raw responses and terminal status are retained. After all eligible parallel role votes terminate, the deterministic aggregator applies the frozen risk/action, evidence, approval, and tie-break rules without an additional model call.

Table 5 summarizes the bundled configuration factors. The listed evidence caps are the runner's historical evidence-selection budgets, not complete prompt-token counts. Every stored role vote has its own maximum output length of 4096 tokens. Consequently, K changes the role set, each role's visible slice, the number of scheduled generations, and the possible inputs to $\mathcal{A}$; Table 5 is not a one-factor agent-count intervention.

Table 5: Configuration-factor mapping.

K	Fixed Role Set $\mathcal{R}_K$	Evidence Cap/Role	Votes/Group	Final Combiner
1	Generalist	3000	1	Deterministic $\mathcal{A}$
5	Manager, log, asset, threat, verifier	600	5	Deterministic $\mathcal{A}$
10	K = 5 roles plus simulation, human-gate, redundant-log, redundant-threat, speculative-hypothesis	300	10	Deterministic $\mathcal{A}$

The role design is a fixed-source parallel-vote design. K = 1 uses a generalist role. K = 5 uses a manager plus log, asset, threat, and verifier roles. K = 10 adds simulation, human-gate, redundant-log, redundant-threat, and speculative-hypothesis roles. This design tests whether adding parallel role votes improves evidence-supported triage under a deterministic scorer. There is no multi-turn interaction, planning order, or tool-mediated data transfer among agents. This choice measures the complete configuration produced when K and its role views change together. It does not test a sequential planner-executor-verifier workflow, a dynamic tool-query workflow, or a patch-remediation workflow. Those are different multi-agent designs and are not inferred from the present results.

The selected routes are not exhaustive. They are used as calibrated operating points for the same instrument. Additional proprietary or open-weight frontier models can be added later only if they are rerun under the same output contract, seeds, token cap, and parse-retry policy.

Reproduction is anchored to the frozen route configuration, role-prompt templates, output schema, source-alert inputs, raw JSONL traces, scorer configuration, aggregation code, and SHA256 manifests. A clean-directory reproduction must first verify those hashes, reconstruct the 45 route–dataset–seed runs and 135 K files, confirm role counts and source signatures, replay deterministic aggregation, and only then regenerate the metric tables and figures. Operating-system, Python/runtime, local-GPU, endpoint, model-route, seed, token, retry, and package-version records are included in the revision artifact map. This sequence fails closed on a missing input, hash mismatch, schema mismatch, group mismatch, or incomplete route grid.

5.5 Same-Temperature Repeated-Inference Control

To distinguish repeated generation from the bundled role-conditioned pipeline, we add a single-generalist control at K = 5 and K = 10. For a given source group and K, all K control requests use the same byte-identical common evidence view and the historical generalist prompt. The control keeps the model route, dataset, seed-indexed source groups, K, output schema, output-token cap, provider-specific generation settings, and bounded retry policy aligned with the role comparison. Both sides are combined by the same neutral deterministic plurality aggregator and are evaluated by the same primary scorer.

The control covers three routes, five datasets, three seeds, and two K values, yielding 90 paired model–dataset–seed–K cells. Each route–K stratum contains 84 seed-indexed source-group rows: 28 per seed, with dataset counts of 6, 6, 4, 6, and 6. The full control therefore contains $3 \times 84 \times (5 + 10) = 3780$ terminal generalist outputs and 504 aggregated control units, each paired with the corresponding historical role-conditioned unit. Intermediate failed attempts are retained for audit but do not become additional scientific outputs.

This is a same-temperature repeated-inference control rather than a canonical high-temperature self-consistency experiment. It matches the request semantics of the historical routes. It also does not make the control and role prompts identical: the control uses one common view, whereas the historical pipeline uses role-conditioned views. The paired comparison can therefore determine whether the role-conditioned pipeline consistently outperforms repeated generalist inference under the tested regime, but it cannot estimate a pure causal effect of role specialization.

5.6 Artifact Records

Outputs are stored as JSONL records, summary JSON files, comma-separated values (CSV) files, and provider logs. For reproducibility accounting, the calibrated matrix contains 45 complete model–dataset–seed cells and 4032 logical role-vote records: 1344 per model route. Two DeepSeek parse regenerations yield 4034 stored generation-attempt records. The historical transport layer did not retain every lower-level transport retry, so an exact HTTP/API request count cannot be reconstructed. None of these operational counts is an independent sample size. No final model–dataset–seed cell is missing from the reported matrix. The artifact package also contains a small validation sample that maps selected outputs to their visible evidence and hidden scoring keys. This sample is intended for independent inspection of the scoring rules. It should not be read as a completed human agreement study.

The repeated-inference control is recorded separately. It contains 3780 terminal logical outputs, 3792 attempt-final records, 504 aggregated control units, and 504 matched historical role units. Intermediate failed

attempts remain in the audit trail but are not counted as additional scientific observations. The paired analysis produces 90 model–dataset–seed–K cells.

In line with the Tech Science Press artificial-intelligence disclosure policy, material AI assistance in manuscript preparation, code inspection, and deterministic revision analysis is disclosed in the Acknowledgment. All model-generated experimental records remain identified separately as research-subject outputs, and every reported numerical result is derived from retained traces and author-reviewed deterministic scripts.

6 Results

6.1 Audit Accounting

The calibrated comparison yields fifteen model–dataset rows across five datasets and three model routes. Seeds 11, 22, and 33 each index a context-stratified sample of 28 groups: 6/6/4/6/6 for AIT-ADS, LANL, CSIC2010, UNSW-NB15, and CICIDS2017. The stored traces therefore contain 84 seed-indexed group observations but only 54 distinct alert-set signatures, with cross-seed overlap. They are not 84 statistically independent samples or repeated stochastic generations on one fixed set. The raw-first corrected pipeline discovered 45 unique route–dataset–seed runs, 135 raw K files, and 756 final outputs. It revalidated every source group, role set, deterministic aggregate, and token total before scoring.

6.2 Deterministic Rule Reference

Before comparing model routes, we add a zero-call deterministic reference over the same source groups. The rule retains high-truth alerts, enforces approval on destructive actions, and suppresses low-truth alerts only when the frozen detector hints permit it. Over both the 84 seed-indexed groups and the 54 signature-deduplicated groups, the component values are CAR = 1.0000, FPRR = 0.1290, ACR = 1.0000, ESR = 1.0000, Q = 0.7823, and UVD_{all} = 0. UCR, R, S, and best-K are not computed because this rule has no explicit claim-producing role votes.

This is an internal operational reference, not an independent external baseline. Detector hints agree with truth for 461 of 537 alerts and for the maximum-risk label in 63 of 84 groups. Only 40 of 310 low-truth alerts avoid human triage; 270 are still promoted. The reference therefore checks that the component scorer and action-routing rules are executable, but it neither establishes detector generalization nor supports a claim that RACE-SOC outperforms a traditional method.

Table 6 reports the corrected PRIMARY best-K choices and mean selected-K scores for the three model routes [40–42] across the three sampling seeds.

Table 6: Corrected three-seed PRIMARY best-K route audit.

Model	Dataset	Seed 11	Seed 22	Seed 33	$\overline{S}_{K^*}$	Same K across Seeds
Qwen3.5-9B	AIT-ADS	5	1	1	0.8013	No
Qwen3.5-9B	LANL	5	5	5†	1.0000	Yes
Qwen3.5-9B	CSIC2010	5†	10	10	0.6250	No
Qwen3.5-9B	UNSW-NB15	5†	5†	5	0.7553	Yes
Qwen3.5-9B	CICIDS2017	5†	5†	5†	0.8323	Yes
GLM-5.2	AIT-ADS	1	1	1	0.6388	Yes
GLM-5.2	LANL	5	10	10	0.8175	No

(Continued)

Table 6 (continued)

Model	Dataset	Seed 11	Seed 22	Seed 33	$\overline{S}_{K^*}$	Same K across Seeds
GLM-5.2	CSIC2010	10	1	1	0.0250	No
GLM-5.2	UNSW-NB15	10	10	5	0.8429	No
GLM-5.2	CICIDS2017	10	10	10	0.6772	Yes
DeepSeek-v4-pro	AIT-ADS	1	5	1	0.6137	No
DeepSeek-v4-pro	LANL	1	10	10	0.7846	No
DeepSeek-v4-pro	CSIC2010	10	10	10	−0.0447	Yes
DeepSeek-v4-pro	UNSW-NB15	10	10	10	0.7517	Yes
DeepSeek-v4-pro	CICIDS2017	1	1	1	0.6415	Yes

Note: Scores and selections are reported at $\lambda = 1$. [†]The seven marked cells have K = 5 and K = 10 co-best within 10^{-12}; K = 5 is the preregistered operational choice after the cost-first tie-break. $\overline{S}_{K^*}$ averages the score at each seed's operational choice. Seeds 11/22/33 index context-stratified group samples, not stochastic generations on one fixed set of units. The table is descriptive and does not support independent-sample inference or a general model leaderboard.

6.3 Context-Dependent Best-K Behavior

The corrected all-K comparison materially changes the original result. At $\lambda = 1$, the operational selections are K = 1/5/10 in 13/14/18 of the 45 route–dataset–seed cells. Seven cells have an exact K = 5/K = 10 co-best set; the cost-first rule selects K = 5 operationally. In exclusive-winner form the result is 13/7/18 plus seven ties, while co-best membership is 13/14/25. These three views are reported together so that the headline is not created by hiding or double-counting ties.

The global macro mean PRIMARY scores over the fifteen route–dataset cells are 0.5465, 0.5497, and 0.5996 for K = 1, K = 5, and K = 10, respectively. These are descriptive averages, not a universal ordering: Table 6 shows different choices across routes, sources, and seed-indexed samples. Only eight of fifteen route–dataset pairs have the same operational K across all three seeds. The corrected result therefore strengthens the paper's scheduling interpretation while overturning the original apparent dominance of K = 1.

Fig. 2 visualizes the best-K net scores from Table 6 by model and dataset, making the non-uniformity of the selected team size visible without changing the underlying audit values.

Fig. 3 complements Fig. 2 by showing how the measured net score moves across the three fixed K settings. The connected lines are not optimised optima. They only help compare the observed K = 1, K = 5, and K = 10 configurations.

The three-seed extension adds a sampling-sensitivity audit but does not raise the pilot to an inferential benchmark. Eight of fifteen route–dataset cells have the same operational K for all three seeds. Because the seeds resample and reorder context groups and share 54 distinct signatures across 84 seed-indexed observations, this is protocol sensitivity evidence rather than population-level statistical confidence.

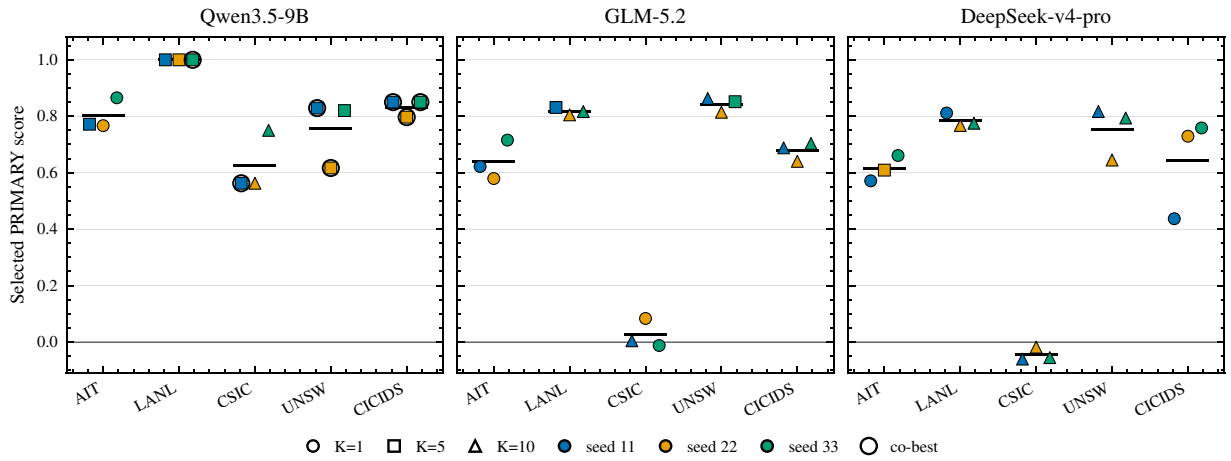


Figure 2: Corrected PRIMARY selected-K score for every route–dataset–seed cell at lambda = 1. A dagger in Table 6 identifies co-best K = 5/K = 10 cells resolved operationally by the preregistered cost-first rule. Seeds are context-sampling indices, not independent population replicates. ESR is syntactic identifier grounding and ACR is branch-dependent; points support within-source protocol comparison, not semantic or cross-dataset model superiority.

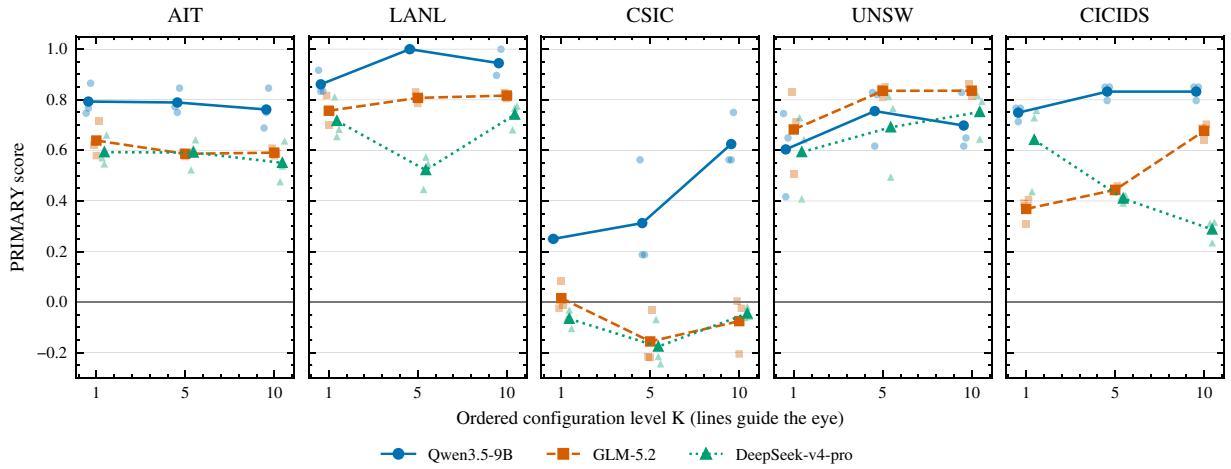


Figure 3: Corrected PRIMARY net-score trajectories over the measured K settings. Dark markers and connected lines show the mean across the three sampling seeds at each K; faint points show the individual sampling-seed scores. Lines connect only K = 1, K = 5, and K = 10 and are visual aids, not interpolations of a continuous optimum.

6.4 Net Scores under the Present Scorer

Net score is the risk-adjusted value $S = Q - \lambda R$. A higher score is better because it reflects stronger evidence-supported utility after the primary all-K failures have been subtracted. A larger role set may increase both utility and risk, so the separate quality and risk axes remain essential even when a net value is used for operational selection. Fig. 4 therefore exposes both components rather than treating S as a sufficient description of model capability.

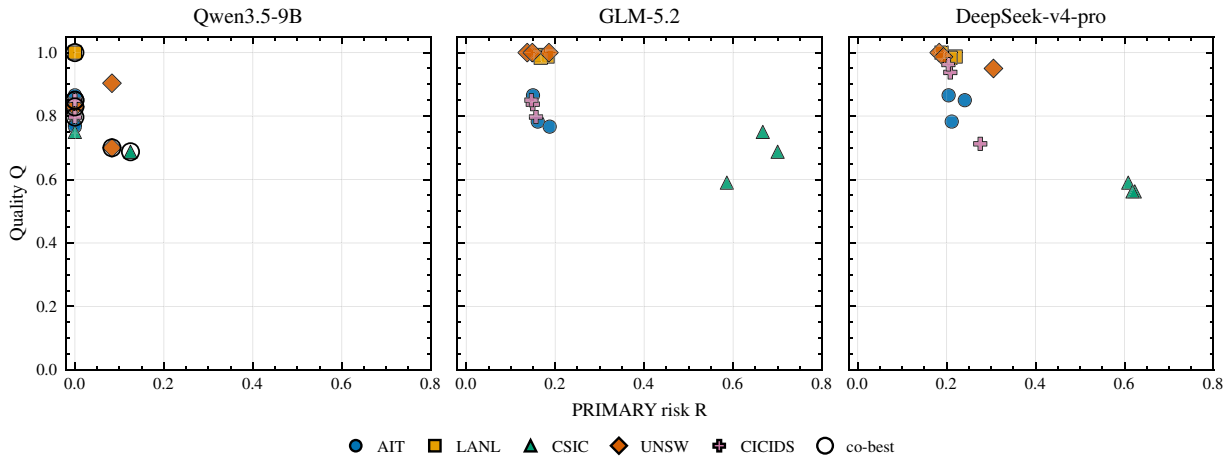


Figure 4: Quality-failure trade-off at the corrected operational K for each route–dataset–seed cell. Net score is quality minus PRIMARY risk at lambda = 1. Because ESR is syntactic and ACR depends on the applicable action branch, cross-dataset distances are descriptive rather than calibrated semantic performance differences.

The corrected scorer no longer mixes an unbounded violation density with bounded risk terms. UVD_{all} uses all scored outputs as its denominator and counts each unsafe output at most once. This correction substantially redistributes the operational best-K choices. It does not, however, eliminate every low-score source pattern. Under the PRIMARY definition, CSIC2010 remains the lowest selected mean score for all three routes in Table 6 (0.6250 for Qwen, 0.0250 for GLM, and −0.0447 for DeepSeek). We therefore retain CSIC2010 only as a within-protocol cautionary observation. The explanation has changed: the pattern cannot be attributed to an unbounded UVD term, and with four groups per seed it does not establish a general property of web-request triage, the dataset, or any model.

Figs. 4 and 5 provide two checks on this interpretation. Fig. 4 divides the selected best-K points into quality and failure components, while Fig. 5 shows which best-K choices remain stable under different risk penalty values.

6.5 Lambda Sensitivity

Lambda is the risk-aversion coefficient in $S = Q - \lambda R$. A larger lambda indicates that the same failure rate receives a higher cost. The lambda sweep over $\{0.25, 0.5, 1, 2, 4\}$ checks whether the best-K choices change when the reference selection of lambda = 1 for the main tables is varied. Table 7 reports the corrected operational sequence. Nine of fifteen route–dataset pairs remain stable over the five lambda values. Six are lambda-sensitive, and the full corrected audit is also classified as risk-definition-sensitive when the prespecified legacy, bounded, saturating, macro-UCR, and outcome-risk variants are compared. The result is therefore labelled BOTH_SENSITIVE. Lambda = 1 is a transparent reference setting, not a fitted or universal SOC preference.

6.6 Pareto Frontier and Offline Replay

This analysis asks whether the recorded fixed agent-count runs contain information for future instance-level scheduling. A Pareto frontier keeps the configurations that are not worse on quality, failure, token use at the same time. Historical latency is excluded because the raw traces do not identify per-group inference time and several batch rows contain cache-read timing. An offline replay policy then simulates choosing among the already measured agent counts without any new model calls. The analysis matters because the practical

question is not which fixed K wins globally, but whether the trace contains signals that help decide when a larger role set is worth its risk and cost.

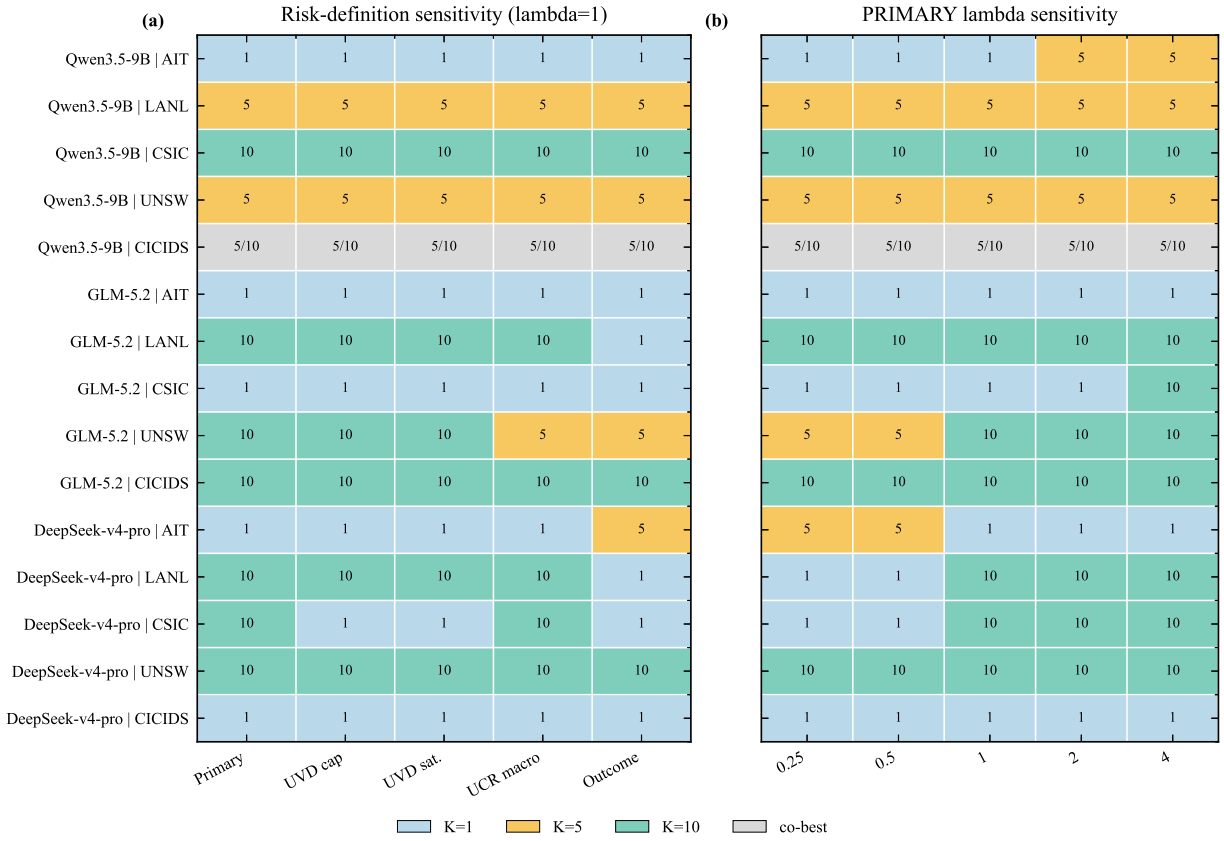


Figure 5: Lambda sensitivity and risk-definition audit for corrected best-K choices. The matrix separates risk-weight sensitivity from changes caused by alternative prespecified risk definitions.

Table 7: Corrected PRIMARY best-*K* sensitivity.

Model	Dataset	Operational Best-*K* Sequence	Stable
Qwen3.5-9B	AIT-ADS	1, 1, 1, 5, 5	No
Qwen3.5-9B	LANL	5, 5, 5, 5, 5	Yes
Qwen3.5-9B	CSIC2010	10, 10, 10, 10, 10	Yes
Qwen3.5-9B	UNSW-NB15	5, 5, 5, 5, 5	Yes
Qwen3.5-9B	CICIDS2017	$5^{\dagger}$, $5^{\dagger}$, $5^{\dagger}$, $5^{\dagger}$, $5^{\dagger}$	Yes
GLM-5.2	AIT-ADS	1, 1, 1, 1, 1	Yes
GLM-5.2	LANL	10, 10, 10, 10, 10	Yes
GLM-5.2	CSIC2010	1, 1, 1, 1, 10	No
GLM-5.2	UNSW-NB15	5, 5, 10, 10, 10	No
GLM-5.2	CICIDS2017	10, 10, 10, 10, 10	Yes
DeepSeek-v4-pro	AIT-ADS	5, 5, 1, 1, 1	No
DeepSeek-v4-pro	LANL	1, 1, 10, 10, 10	No

(Continued)

Table 7 (continued)

Model	Dataset	Operational Best-*K* Sequence	Stable
DeepSeek-v4-pro	CSIC2010	1, 1, 10, 10, 10	No
DeepSeek-v4-pro	UNSW-NB15	10, 10, 10, 10, 10	Yes
DeepSeek-v4-pro	CICIDS2017	1, 1, 1, 1, 1	Yes

Note: The five sequence entries correspond, in order, to $\lambda = 0.25/0.5/1/2/4$. †K = 5 and K = 10 are co-best; the displayed operational choice is K = 5 after the preregistered cost-first tie-break. Stability refers to that operational sequence and does not imply invariance to the risk definition.

The fixed-K traces support a descriptive offline replay because all three K settings were executed under the same model route, seed, scorer, and triage-unit construction. Tokens are reconstructed directly from raw role-vote records on a per-group basis rather than from the legacy cumulative K fields. Each leave-one-seed-out fold chooses both the fixed baseline and replay rule from the two training seeds only; source-aware rules are constrained to the matching dataset/source.

For the corrected Pareto audit, a fixed-K point is dominated if another measured K for the same route, dataset, and seed has at least as high quality and no higher PRIMARY risk or tokens per group, with a strict improvement in at least one dimension. Across 135 measured fixed-K points, 41 are dominated. Twelve of 45 cells have a single-K frontier; the remainder retain multiple non-dominated choices. This confirms that some settings are inefficient under the corrected scorer, while no single K exhausts the measured frontier.

For offline replay, we compare three policies over the context-pressure strata already produced by the experiment. The baseline is the best single fixed K for each model route. The source-aware replay policy uses the other two seeds to choose K for each model, dataset/source, and context-pressure stratum, then evaluates that choice on the held-out seed. The constrained source-aware policy changes K only when training seeds show higher net score without higher failure. The oracle selects the best K after seeing the held-out stratum outcome and is used only to measure the available gap.

Table 8 summarizes the held-out replay results.

Table 8: Corrected leave-one-seed-out replay over recorded fixed-*K* traces.

Policy Scope	Strata	Fixed *S*	Replay *S*	Oracle *S*	Gap rec.	Replay *R*	Tokens/ Group	Δ Fixed	Latency
Pressure-only	117	0.5494	0.5639	0.6232	19.6%	0.2204	5232.0	−30.3%	N/A
Source-aware	117	0.5494	0.5811	0.6232	43.0%	0.2116	6029.8	−19.7%	N/A
Constrained source-aware	117	0.5494	0.5661	0.6232	22.6%	0.2031	7171.9	−4.5%	N/A

Table 8 gives a bounded reading. The source-aware rule recovers 43.0% of the corrected oracle gap and uses 19.7% fewer raw-vote tokens per group than the training-fold fixed baseline. Pressure-only and constrained source-aware rules recover 19.6% and 22.6%, respectively. These values are exploratory rather than independent generalization estimates: training and held-out seed samples overlap in source-group signatures. Historical latency is reported as N/A and is not used for dominance or policy selection. The replay

therefore motivates future risk-constrained scheduling without claiming that these simple rules are ready for online deployment.

6.7 Repeated Inference vs. the Role-Conditioned Pipeline

Table 9 reports the paired control that separates sampling count from the role-conditioned configuration. Each difference is the same-temperature repeated-inference control minus the matched historical role pipeline, averaged over five datasets and three seeds. A positive ΔS favors repeated generalist inference. A negative ΔR means that the control has lower measured risk.

Table 9: Paired repeated-inference control at K = 5 and K = 10.

Model Route	K	ΔQ	ΔR	ΔS at $\lambda = 1$
Qwen3.5-9B	5	0.053890	−0.075000	0.128890
Qwen3.5-9B	10	0.075056	−0.050000	0.125056
GLM-5.2	5	0.037966	−0.023998	0.061965
GLM-5.2	10	−0.042937	−0.003923	−0.039014
DeepSeek-v4-pro	5	0.119452	−0.075517	0.194969
DeepSeek-v4-pro	10	0.033174	−0.015266	0.048440

The repeated-inference control has the higher mean net score in five of the six model–K strata. The exception is GLM-5.2 at K = 10, where the role pipeline is higher by 0.039014. This mixed result directly changes the interpretation of K. Under the tested design, the role wrapper does not provide a consistent advantage over repeating a generalist response. The GLM-5.2 K = 10 counterexample also prevents the opposite universal claim that repeated inference is always preferable.

Three additional analyses test whether this conclusion is an artifact of repeated source groups, aggregation ties, or unequal evidence exposure. First, the 84 seed-indexed rows contain 54 distinct alert-set signatures. Equal weighting of those signatures preserves the direction of all ten group-level endpoints in all six model–K strata. Second, 76 of the 504 paired units contain a plurality tie on at least one side. Excluding each such pair and rescoring preserves all six net-score directions, including the five-positive, one-negative pattern. Third, the total visible evidence differs by only +79 tokens for K = 5 and −689 tokens for K = 10 when the control is compared with the role views over 84 seed-indexed groups. Across the 90 paired run cells, the signed exposure difference has a Pearson correlation of −0.0475 with ΔS ($R^2 = 0.0023$). These are descriptive robustness checks. They show that no one of the three audited factors is an obvious single explanation for the observed pattern, but they do not eliminate unmeasured prompt or provider-time differences.

6.8 Role-Design Boundary

This study evaluates one bundled role design. All roles are assigned to the same source group, receive role-conditioned prompt views, and submit structured votes independently. A deterministic rule aggregator, rather than a second manager-model call, produces the final output. The paired repeated-inference control shows that this wrapper does not consistently improve the primary net score: the control is higher in five of six model–K strata, while the role pipeline is higher for GLM-5.2 at K = 10. RACE-SOC should therefore be viewed as a tool for auditing configuration boundaries rather than evidence that role specialization or a particular K is generally optimal.

Interactive role designs remain important next steps. A planner-executor-verifier sequence could assign different functions to different roles. A dynamic SOC workflow could query asset inventories, playbooks, and

containment policies. A remediation workflow would need to reward early patch completion and penalise delayed repair. These are not failures of the current experiment. They are different workflows that can be evaluated by RACE-SOC after their traces have been logged under the same structured output contract and scoring rules.

7 Discussion and Validity Boundaries

The calibrated pilot supports the central idea of RACE-SOC, but it remains a protocol demonstration rather than a statistical benchmark. Each seed contains 28 sampled groups (6/6/4/6/6), while the three seeds yield 84 seed-indexed observations but only 54 distinct alert-set signatures with overlap. Within this limited scope, changing K changes a bundle of factors: role composition, prompt views, generation count, and aggregation inputs. RACE-SOC makes the resulting utility and risk trade-off measurable under a deterministic rule set. Its main finding is not that fewer or more agents are always better. It is that the measured configuration boundary is conditional on the model, data source, sampling seed, scorer, and protocol.

The repeated-inference control is central to this interpretation. A single generalist repeated K times obtains a higher mean primary net score than the historical role pipeline in five of six model–K strata. The role pipeline remains higher for GLM-5.2 at K = 10. Unique-signature weighting and exclusion of plurality ties preserve all six directions, and visible-token exposure has almost no pooled linear association with the paired net-score difference. These checks strengthen the conclusion that the role wrapper has no consistent advantage in this experiment. They do not isolate role specialization as a causal factor because the common generalist view and role-conditioned prompt views remain different.

The Reviewer 2 corrections are consequential rather than cosmetic. Once ADR/FCR are removed from the all-K score and legacy UVD is replaced by bounded all-output incidence, the operational K = 1/5/10 counts change from the legacy 36/4/5 to 13/14/18, including seven K = 5/K = 10 co-best cells. Only eight of fifteen route–dataset pairs select the same operational K across the three sampling seeds, and nine of fifteen are stable over the tested lambda grid. The old apparent K = 1 dominance was therefore partly a scoring artifact and is not retained as a scientific conclusion.

This result points to scheduling rather than to a universal agent count. A useful SOC scheduler should select the smallest coordination structure that preserves evidence support and action safety for the current triage unit. It should spend additional tokens only when larger role sets are likely to close a quality gap. The offline replay analysis supports this direction. Source-aware replay recovers 43.0% of the corrected oracle gap in the overlapping-seed trace, but this is an exploratory scheduling signal rather than independent generalization. RACE-SOC therefore provides audited traces and scheduler signals rather than a finished online scheduler.

Several boundaries should guide interpretation. First, the scorer is one operationalization of SOC triage quality. It uses syntactic evidence grounding, a branching action-consistency metric, bounded all-output unsafe incidence, micro claim aggregation, and unweighted primary quality and risk means. Prespecified alternative definitions do reorder some operational choices, so the result is classified as BOTH_SENSITIVE. Second, the realized unit count is small and the seed samples overlap. The three-seed audit measures sensitivity to context-stratified resampling and prompt ordering; it does not provide population-level confidence.

Third, the evaluated role design is fixed-source parallel voting. The same source groups are used across paired configurations, but the actual prompt fields and allocations differ between common and role-conditioned views. The repeated-inference control also uses each provider's historical same-temperature

request regime rather than a canonical high-temperature self-consistency setting. It does not evaluate sequential planner-executor-verifier workflows, dynamic tool-query SOC workflows, or remediation workflows that must generate and test patches. Fourth, the datasets are public sources converted into bounded triage units. They do not reproduce all constraints of a live SOC, including incomplete telemetry, organization-specific playbooks, analyst feedback, and adversarial prompt manipulation. Fifth, the component threat model is only partially measured: foundation-model backdoors, malicious instructions, compromised dependencies, and provider supply-chain substitution remain outside the benign-trace coordination audit. Sixth, per-group inference latency is not identifiable from the stored historical traces; latency is therefore excluded from the corrected Pareto and replay conclusions.

These boundaries define the appropriate use of the results. RACE-SOC is a reproducible evaluation instrument for multi-agent SOC triage under a fixed source context, meaning that no compared configuration retrieves extra source material during the run. Within that setting, it exposes where role-conditioned configurations help, where repeated generalist inference is sufficient, and which recorded signals can motivate future risk-adaptive scheduling.

8 Conclusion

This paper introduced RACE-SOC, a risk-constrained evaluation protocol and reproducible pilot for multi-agent SOC alert triage, in which agent count is used as a coordination-risk stress variable and a deterministic, code-anchored scorer estimates the boundary where extra agents remain useful. In the calibrated strong-model matrix, Qwen3.5-9B, GLM-5.2, and DeepSeek-v4-pro are compared across five public datasets, three seeds, and agent counts $K \in \{1, 5, 10\}$ under a 4096-token output cap. The best fixed-K configuration is conditional rather than monotonic. Under the corrected K-invariant PRIMARY score, the operational K = 1/5/10 counts are 13/14/18; seven K = 5/K = 10 cells are co-best and are resolved by the preregistered cost-first tie-break. Eight of fifteen route–dataset pairs select the same operational K across the three sampling seeds, and nine of fifteen are stable across the tested lambda values. The result is sensitive to both risk definition and lambda, so the legacy 36/4/5 count is retained only as an audit replay. Corrected Pareto analysis finds 41 of 135 fixed-K points dominated and 12 of 45 cells with a single-K frontier. Source-aware offline replay recovers 43.0% of the corrected oracle gap in the overlapping-seed trace while using 19.7% fewer raw-vote tokens per group than its training-fold fixed baseline. This replay is exploratory, not an independent generalization estimate, and historical latency is excluded because it cannot be reconstructed from the stored traces.

The paired repeated-inference control further changes what these fixed-K results mean. Repeated generalist inference has the higher mean primary net score in five of six model–K strata, while the role-conditioned pipeline is higher for GLM-5.2 at K = 10. The six directions remain stable after unique-signature weighting and tie exclusion. The evidence therefore does not support a general role-specialization benefit under this design.

These findings support a bounded conclusion. RACE-SOC does not argue that one model is generally better than another, or that a single-agent SOC design is universally optimal. It argues that multi-agent coordination should be evaluated as a risk-cost frontier, because added roles can improve evidence coverage in some settings while adding unsupported claims, disagreement, false consensus, or token cost in others. Existing fixed-K traces contain scheduling signal, but simple source and pressure rules are not sufficient for a robust online scheduler. Every reported quantity is recomputable from recorded artifacts under a documented rule set. The practical position is that multi-agent SOC systems should be scheduled by measured evidence support, failure penalties, and cost rather than by the assumption that more agents are inherently safer. Latency must be measured prospectively rather than inferred from the present traces.

Acknowledgement: During preparation of this revision, the authors used OpenAI ChatGPT and Codex to assist language editing, code inspection, deterministic analysis scripting, and preparation of revision materials. The authors reviewed and revised the assisted content; verified the cited sources, retained experimental traces, calculations, tables, and figures; and accept full responsibility for the article. The evaluated Qwen, GLM, and DeepSeek systems are research subjects and are described separately in the Methods. No AI tool was permitted to invent, alter, or replace experimental data.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: Conceptualization, Yi Lv. Methodology, Yi Lv and Cheng Wang. Software, Yi Lv. Validation, Cheng Wang. Formal analysis, Yi Lv. Investigation, Yi Lv. Resources, Yi Lv. Data curation, Yi Lv. Writing original draft preparation, Yi Lv. Writing review and editing, Cheng Wang. Visualization, Yi Lv. Supervision, Cheng Wang. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The study uses public cybersecurity datasets cited in the Experimental Setup. The existing Zenodo record (https://doi.org/10.5281/zenodo.21264368) and GitHub release v2.0.0 (https://github.com/estigeralynhbettin-eng/RACE-SOC-CMC-artifact/releases/tag/v2.0.0) preserve the legacy submitted analysis and are retained for audit continuity; they do not contain the complete corrected revision evidence and must not be used to reproduce the revised PRIMARY conclusions. A clean-room revision artifact accompanies the resubmission package and contains the source-alert inputs, allowlisted raw traces, corrected scorer and aggregator, repeated-inference control, deterministic rule reference, generated table/figure data, replay outputs, environment records, and SHA256 manifests. Public replacement of the legacy release will occur only after author approval; the local revision package is the authoritative review artifact for this resubmission.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang J, Zhou C, Yi Y, Sun Q, Yan T, Qi F. ChatSOC: a large language model powered autonomous agent for security operations center. In: Proceedings of the ISGC2025; 2025 Mar 16–21; Taipei, Taiwan. doi:10.22323/1.488.0009.
2. Boffa M, Drago I, Mellia M, Vassio L, Giordano D, Valentim R, et al. LogPrécis: unleashing language models for automated malicious log analysis. Comput Secur. 2024;141(10):103805. doi:10.1016/j.cose.2024.103805.
3. Chen Y, Cui M, Wang D, Cao Y, Yang P, Jiang B, et al. A survey of large language models for cyber threat detection. Comput Secur. 2024;145(9):104016. doi:10.1016/j.cose.2024.104016.
4. Xu H, Wang S, Li N, Wang K, Zhao Y, Chen K, et al. Large language models for cyber security: a systematic literature review. ACM Trans Softw Eng Methodol. 2025;8:3769676. doi:10.1145/3769676.
5. Freitas S, Kalajdjieski J, Gharib A, McCann R. AI-driven guided response for security operation centers with microsoft copilot for security. In: Proceedings of the Companion Proceedings of the ACM on Web Conference 2025; 2025 May 13–17; Sydney, Australia. New York, NY, USA: ACM; 2025. p. 191–200. doi:10.1145/3701716.3715209.
6. Binbeshr F, Imam M, Ghaleb M, Hamdan M, Rahim MA, Hammoudeh M. The rise of cognitive SOCs: a systematic literature review on AI approaches. IEEE Open J Comput Soc. 2025;6(2):360–79. doi:10.1109/ojcs.2025.3536800.
7. Qian C, Liu W, Liu H, Chen N, Dang Y, Li J, et al. ChatDev: communicative agents for software development. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); 2024 Aug 11–16; Bangkok, Thailand. p. 15174–86. doi:10.18653/v1/2024.acl-long.810.
8. Hu Y, Zou F, Han J, Sun X, Wang Y. LLM-TIKG: threat intelligence knowledge graph construction utilizing large language model. Comput Secur. 2024;145:103999.
9. Belcastro L, Carlucci C, Cosentino C, Liò P, Marozzo F. Enhancing network security using knowledge graphs and large language models for explainable threat detection. Future Gener Comput Syst. 2026;176(7):108160. doi:10.1016/j.future.2025.108160.

10. Baruwal Chhetri M, Tariq S, Singh R, Jalalvand F, Paris C, Nepal S. Towards human-AI teaming to mitigate alert fatigue in security operations centres. ACM Trans Internet Technol. 2024;24(3):1–22. doi:10.1145/3670009.
11. Ban T, Samuel N, Takahashi T, Inoue D. Combat security alert fatigue with AI-assisted techniques. In: Proceedings of the Cyber Security Experimentation and Test Workshop; 2021 Aug 9; Virtual. p. 9–16. doi:10.1145/3474718.3474723.
12. Forsberg J, Frantti T. Technical performance metrics of a security operations center. Comput Secur. 2023;135:103529. doi:10.1016/j.cose.2023.103529.
13. Bhatt S, Manadhata PK, Zomlot L. The operational role of security information and event management systems. IEEE Secur Priv. 2014;12(5):35–41. doi:10.1109/msp.2014.103.
14. Ghadermazi J, Shah A, Jajodia S. A machine learning and optimization framework for efficient alert management in a cybersecurity operations center. Digit Threat. 2024;5(2):1–23. doi:10.1145/3644393.
15. Chavali L, Krishnan A, Saxena P, Mitra B, Sreevallabh Chivukula A. Off-policy actor-critic deep reinforcement learning methods for alert prioritization in intrusion detection systems. Comput Secur. 2024;142(3):103854. doi:10.1016/j.cose.2024.103854.
16. Wang X, Yang X, Liang X, Zhang X, Zhang W, Gong X. Combating alert fatigue with AlertPro: context-aware alert prioritization using reinforcement learning for multi-step attack detection. Comput Secur. 2024;137(1):103583. doi:10.1016/j.cose.2023.103583.
17. Landauer M, Skopik F, Wurzenberger M, Rauber A. Dealing with security alert flooding: using machine learning for domain-independent alert aggregation. ACM Trans Priv Secur. 2022;25(3):1–36. doi:10.1145/3510581.
18. Cuppens F, Miege A. Alert correlation in a cooperative intrusion detection framework. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy; 2002 May 12–15; Berkeley, CA, USA. p. 202–15.
19. Hassan WU, Guo S, Li D, Chen Z, Jee K, Li Z, et al. NoDoze: combatting threat alert fatigue with automated provenance triage. In: Proceedings of the 2019 Network and Distributed System Security Symposium; 2019 Feb 24–27; San Diego, CA, USA. doi:10.14722/ndss.2019.23349.
20. Hassan WU, Noureddine MA, Datta P, Bates A. OmegaLog: high-fidelity attack investigation via transparent multi-layer log analysis. In: Proceedings of the NDSS 2020; 2020 Feb 23–26; San Diego, CA, USA.
21. Han X, Pasquier T, Bates A, Mickens J, Seltzer M. Unicorn: runtime provenance-based detector for advanced persistent threats. In: Proceedings of the 2020 Network and Distributed System Security Symposium; 2020 Feb 23–26; San Diego, CA, USA. doi:10.14722/ndss.2020.24046.
22. Goyal A, Wang G, Bates A. R-CAID: embedding root cause analysis within provenance-based intrusion detection. In: Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP); 2024 May 20–23; San Francisco, CA, USA. p. 3515–32.
23. Du M, Li F, Zheng G, Srikumar V. DeepLog: anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security; 2017 Oct 30–Nov 3; Dallas, TX, USA. p. 1285–98. doi:10.1145/3133956.3134015.
24. Guo H, Yuan S, Wu X. LogBERT: log anomaly detection via BERT. In: Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN); 2021 Jul 18–22; Shenzhen, China. p. 1–8.
25. Mirsky Y, Doitshman T, Elovici Y. Shabtai A. Kitsune: An ensemble of autoencoders for online network intrusion detection. In: Proceedings of the 2018 Network and Distributed System Security Symposium; 2018 Feb 18–21; San Diego, CA, USA. doi:10.14722/ndss.2018.23204.
26. Landauer M, Skopik F, Wurzenberger M. Introducing a new alert data set for multi-step attack analysis. In: Proceedings of the 17th Cyber Security Experimentation and Test Workshop; 2024 Aug 13; Philadelphia, PA, USA. p. 41–53. doi:10.1145/3675741.3675748.
27. Kent AD. Comprehensive, multi-source cyber-security events data set. Los Alamos, NM, USA: Los Alamos National Laboratory; 2015.
28. Impact Cyber Trust. HTTP DATASET CSIC 2010 [Internet]. [cited 2026 Jul 5]. Available from: https://www.impactcybertrust.org/dataset_view?idDataset=940.

29. Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS); 2015 Nov 10–12; Canberra, Australia. p. 1–6. doi:10.1109/milcis.2015.7348942.
30. Sharafaldin I, Habibi Lashkari A, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy; 2018 Jan 22–24; Funchal, Portugal. p. 108–16.
31. Li G, Hammoud H, Itani H, Khizbullin D, Ghanem B. CAMEL: communicative agents for "mind" exploration of large language model society. Adv Neural Inf Process Syst. 2023;36:51991–2008. doi:10.52202/075280-2264.
32. Wu Q, Bansal G, Zhang J, Wu Y, Li B, Zhu E, et al. AutoGen: enabling Next-Gen LLM applications via multi-agent conversation. arXiv:2308.08155. 2023.
33. Chen W, Su Y, Zuo J, Yang C, Yuan C, Chan CM, et al. AgentVerse: facilitating multi-agent collaboration and exploring emergent behaviors. In: Proceedings of the International Conference on Learning Representations; 2024 May 7–11; Vienna, Austria.
34. Raghavan B, Schneier B. Agentic AI's OODA loop problem. IEEE Secur Priv. 2025;23(6):80–2. doi:10.1109/msec.2025.3604105.
35. Du Y, Li S, Torralba A, Tenenbaum JB, Mordatch I. Improving factuality and reasoning in language models through multiagent debate. In: Proceedings of the 41st International Conference on Machine Learning; 2024 Jul 21–27; Vienna, Austria. p. 11733–63.
36. Liang T, He Z, Jiao W, Wang X, Wang Y, Wang R, et al. Encouraging divergent thinking in large language models through multi-agent debate. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing; 2024 Nov 12–16; Miami, FL, USA. p. 17889–904. doi:10.18653/v1/2024.emnlp-main.992.
37. Li Y, Du Y, Zhang J, Hou L, Grabowski P, Li Y, et al. Improving multi-agent debate with sparse communication topology. In: Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2024; 2024 Nov 12–16; Miami, FL, USA. Stroudsburg, PA, USA: ACL; 2024. p. 7281–94. doi:10.18653/v1/2024.findings-emnlp.427.
38. Zhao S, Tuan LA, Fu J, Wen J, Luo W. Exploring clean label backdoor attacks and defense in language models. IEEE/ACM Trans Audio Speech Lang Process. 2024;32(1):3014–24. doi:10.1109/taslp.2024.3407571.
39. Zhao S, Guo Z, Wu X, Wu X, Jia Y, Xiao L, et al. Protecting your customized LLM systems from backdoored instructions with metacognitive probing. IEEE Trans Inf Forensic Secur. 2026;21:7110–24. doi:10.1109/tifs.2026.3714137.
40. Qwen Team. Qwen3.5-9B model card [Internet]. 2026 [cited 2026 Aug 12]. Available from: https://huggingface.co/Qwen/Qwen3.5-9B.
41. GLM-5 Team. GLM-5.2 model card and GLM-5 technical report. arXiv:2602.15763. 2026.
42. DeepSeek. DeepSeek-V4-Pro models and API documentation [Internet]. 2026 [cited 2026 Aug 12]. Available from: https://api-docs.deepseek.com/quick_start/pricing.