SecLens: Role-Specific Evaluation of LLMs for Security Vulnerability Detection

Subho Halder*[1], Siddharth Saxena[1], Kashinath Kadaba Shrish[1], and Thiyagarajan M[1]

[1]Independent Researchers

Abstract

Existing benchmarks for LLM-based vulnerability detection reduce a model's capability to one number. That number cannot serve the divergent needs of a CISO who prioritizes critical-vulnerability recall, an engineering leader who optimizes for low false-positive rates, or an AI officer who weighs cost against capability. We introduce SecLens-R, a multi-stakeholder evaluation framework built on 35 shared dimensions across 7 measurement categories. Five role-specific weight profiles (CISO, Chief AI Officer, Security Researcher, Head of Engineering, and AI-as-Actor) each select 12–16 dimensions with weights summing to 80, producing a composite Decision Score between 0 and 100. We evaluate 12 frontier models on a dataset of 406 tasks drawn from 93 open-source projects spanning 10 programming languages and 8 OWASP-aligned vulnerability categories, using both Code-in-Prompt (CIP) and Tool-Use (TU) evaluation layers. Decision Scores diverge by up to 31 points across roles for the same model: Qwen3-Coder earns an A (76.3) for Head of Engineering but a D (45.2) for CISO; GPT-5.4 earns an A (76.7) for Head of Engineering but a D (48.4) for CISO. These results confirm that model selection for security vulnerability detection is not a single-objective problem, and that stakeholder-aware evaluation surfaces information that aggregate scores cannot.

Keywords: large language models, security vulnerability detection, benchmark, role-specific evaluation, multi-stakeholder decision framework

1 Introduction

Large language models are increasingly applied to security vulnerability detection [Pearce et al., 2022, Tony et al., 2023, Fang et al., 2024, Sheng et al., 2025]. Models such as GPT-5.4 [OpenAI, 2025], Claude Sonnet 4.6 and Opus 4.6 [Anthropic, 2025], and Gemini 3.x [Google, 2025] can identify, classify, and localize vulnerabilities across diverse programming languages and CWE categories. The pace of benchmark development has accelerated in parallel: from CyberSecEval [Bhatt et al., 2023] and PrimeVul [Ding et al., 2024] in 2023–2024, to SEC-bench [Lee et al., 2025], SecVulEval [Lu et al., 2025], and TOSSS [Damie et al., 2026] in 2025–2026. Organizations now face a practical question: which model should they deploy?

These benchmarks share a critical limitation: they collapse a model's rich performance profile into a single aggregate score. While such scores are useful for leaderboard rankings, they are insufficient for the decisions organizations actually face.

Consider the divergent needs of different organizational stakeholders:

- A **Chief Information Security Officer (CISO)** asks: "Can I trust this model in my security program?" The CISO prioritizes low false-negative rates on critical vulnerabilities, severity-weighted recall, and consistency across CWE categories. A model that excels at injection detection but silently misses authentication bypasses is unacceptable.
- A **Chief AI Officer (CAIO)** asks: "Which model unlocks new capabilities while balancing risk and cost?" The CAIO cares about MCC-per-dollar, autonomous completion rates, and tool-use effectiveness at scale.
- A **Security Researcher** asks: "How deep and reliable is this model's vulnerability reasoning?" Researchers need CWE taxonomy mastery, evidence chain completeness, and reasoning quality on both true positives and false positives.
- A **Head of Engineering** asks: "Will this help or hurt my team's velocity and code quality?" Engineering leaders optimize for high precision (low false-positive rates), fast wall times, low cost per task, and actionable findings with CWE and location.

*Corresponding author. Email: subho.halder@gmail.com

- An **AI-as-Actor** evaluation asks: "Does the agent know what it can and can't do?" This lens evaluates parse reliability, format compliance, error handling, autonomous completion, and graceful degradation under varying task difficulty.

A single benchmark score cannot serve all five perspectives. A model ranked first overall might be the worst choice for a CISO if it misses critical vulnerabilities, or the worst choice for engineering if it generates excessive false positives. Our empirical results confirm this: the same model can score 31 points apart depending on which stakeholder lens is applied.

1.1 Contributions

We make the following contributions:

1. **35 shared evaluation dimensions** across 7 categories, applied through 5 role-specific weight profiles where each role selects 12–16 dimensions with weights summing to 80.
2. **A weighted composite scoring methodology** with dynamic dimension exclusion: when a dimension is unavailable (e.g., tool-use metrics for Code-in-Prompt runs), the denominator adjusts automatically.
3. **Four normalization strategies** with fixed reference caps, eliminating cohort-relative scoring artifacts.
4. **Integration with SecLens**, a two-layer benchmark evaluating LLMs on vulnerability detection using both Code-in-Prompt (CIP) and Tool-Use (TU) paradigms.
5. **Empirical validation across 12 frontier models** on 406 tasks, demonstrating Decision Score divergences of up to 31 points and confirming that leaderboard rank does not predict role-specific rank.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the framework design. Section 4 provides the dimension catalog. Section 5 describes the evaluation methodology. Section 6 details the experimental design. Section 7 presents results and analysis. Section 8 addresses limitations and future work. Section 9 concludes.

2 Related Work

2.1 LLM Security Benchmarks

Several benchmarks evaluate LLM capabilities in security contexts. CyberSecEval [Bhatt et al., 2023] from Meta evaluates both the tendency of LLMs to generate insecure code and their ability to assist in cyberattacks. CyberSecEval 2 [Bhatt et al., 2024] adds prompt injection resistance and code interpreter abuse. CyberSecEval 3 [Wan et al., 2024] extends to offensive capability evaluation. SecEval [Li et al., 2023] covers code generation security across 130 CWE categories. SWE-bench [Jiménez et al., 2024] evaluates models on real-world GitHub issues, though it targets bug-fixing rather than vulnerability detection.

More recent work has sharpened the picture. PrimeVul [Ding et al., 2024] demonstrates that existing benchmarks dramatically overestimate vulnerability detection performance (68% F1 on BigVul drops to 3% on their deduplicated dataset). VulBench [Gao et al., 2023] aggregates CTF and real-world CVEs with root cause annotations. VulDetectBench [Liu et al., 2024b] evaluates 17 models across tasks of increasing difficulty, finding that models achieve 80%+ on binary identification but under 30% on detailed analysis. SecLLMHolmes [Ullah et al., 2024] shows that LLM vulnerability reasoning is non-deterministic and sensitive to variable naming. IRIS [Li et al., 2024] combines LLMs with static analysis to detect 103% more vulnerabilities than CodeQL alone. SAST-Bench [Feiglin and Dar, 2025] evaluates LLM agents on SAST false-positive triage using real CVEs. A recent survey by Sheng et al. [2025] provides a structured overview of LLM architectures, fine-tuning strategies, and evaluation metrics for vulnerability detection.

Three concurrent benchmarks are particularly relevant. SecVulEval [Lu et al., 2025] is the largest CVE-grounded benchmark to date, covering 25,440 C/C++ function samples across 5,867 CVEs with statement-level ground truth; the best model (Claude 3.7 Sonnet) achieves only 23.83% F1, underscoring the difficulty of the task. SEC-bench [Lee et al., 2025] introduces automated CVE reproduction with proof-of-concept generation and patch validation, evaluating LLM agents on 200 verified instances; top models achieve at most 18% PoC generation and 34% patching success. TOSSS [Damie et al., 2026], published concurrently with our work, frames vulnerability detection as binary snippet selection across C/C++ and Java CVEs; model scores range from 0.48 to 0.89.

LLMSecEval [Tony et al., 2023] provides security-relevant code completions based on MITRE CWE scenarios. SecurityEval [Siddiq and Santos, 2022] offers a focused dataset for evaluating LLM-generated code. RepoSim [Zhang et al., 2024] evaluates repository-level vulnerability detection with multi-file contexts.

All of these benchmarks produce single aggregate scores or per-category breakdowns without stakeholder-specific interpretation. None addresses

the organizational decision context in which evaluation results are consumed. Our work is complementary: the SecLens-R scoring layer can consume output from any of these benchmarks, transforming a single leaderboard into five role-specific evaluations.

2.2 Role-Based Evaluation in Software Engineering

The notion that different stakeholders require different evaluation lenses is well-established in software engineering. ISO/IEC 25010 [ISO/IEC, 2011] defines software quality from multiple viewpoints including users, developers, and operators. The Goal-Question-Metric (GQM) paradigm [Basili et al., 1994] formalizes the idea that metrics should be derived from stakeholder goals. In testing, risk-based approaches [Amland, 1999] weight test cases by organizational impact rather than code coverage alone.

In the machine learning evaluation literature, Model Cards [Mitchell et al., 2019] and Datasheets for Datasets [Gebru et al., 2021] advocate for multi-stakeholder transparency, but focus on documentation rather than providing different quantitative scores for different roles. HELM [Liang et al., 2022] evaluates LLMs across multiple scenarios and metrics but does not aggregate by stakeholder role. Chatbot Arena [Chiang et al., 2024] uses crowdsourced preferences to produce Elo ratings, a single-dimensional ranking that our work explicitly moves beyond. AgentBench [Liu et al., 2024a] evaluates LLMs as agents across diverse tasks, relevant to our AI-as-Actor lens but without role-specific scoring.

2.3 Multi-Stakeholder Decision Frameworks

Multi-criteria decision analysis (MCDA) [Velasquez and Hester, 2013] provides formal methods for aggregating multiple evaluation dimensions with role-specific weight vectors. The Analytic Hierarchy Process (AHP) [Saaty, 1990] and TOPSIS [Hwang and Yoon, 1981] are widely used MCDA techniques for technology selection decisions. Recent work on cost-aware LLM evaluation [Sun et al., 2024] demonstrates the practical need for frameworks that consider inference cost alongside quality. Our approach draws on these methods while adapting them to LLM security evaluation: a shared pool of 35 dimensions (rather than per-role dimension sets) simplifies maintenance and enables direct comparison, while role-specific weight vectors preserve stakeholder-specific priorities.

Table 1: Stakeholder roles and their core evaluation questions.

Role	Core Decision Question
CISO	"Can I trust this model in my security program?"
CAIO / Head of AI	"Which model unlocks new capabilities while balancing risk and cost?"
Security Researcher	"Does this model genuinely understand vulnerability mechanics?"
Head of Engineering	"Will this help or hurt my team's velocity and code quality?"
AI as Actor	"Does the agent know what it can and can't do?"

3 Framework Design

3.1 Stakeholder Roles

We define five stakeholder roles, each representing a distinct decision context. Table 1 summarizes the roles and their core decision questions.

CISO. The Chief Information Security Officer is responsible for the organization's security posture, regulatory compliance, and risk management. This lens selects 16 dimensions and allocates 34 of its 80 weight points to Detection (D1/10, D2/8, D3/6, D6/5, D8/5), 18 to Risk & Severity (D28/10, D29/8), and smaller allocations to Coverage, Reasoning, Efficiency, and Robustness. The CISO accepts higher costs and lower throughput in exchange for trustworthy, severity-aware coverage.

CAIO / Head of AI. The Chief AI Officer evaluates models from a strategic capability and efficiency perspective. This lens selects 14 dimensions, distributing weight across Robustness (D31/4, D32/6, D34/10 = 20), Efficiency (D18/5, D20/8, D22/6 = 19), Detection (D1/9, D4/7 = 16), and Tool-Use (D25/5, D26/3, D27/7 = 15). The CAIO values tool-use effectiveness and autonomous completion as indicators of deployment readiness.

Security Researcher. The security researcher requires deep, technically rigorous evaluation. This lens selects 13 dimensions and concentrates 39 weight points on Detection (D1/8, D2/6, D6/12, D7/10, D8/3) and 21 on Reasoning (D14/10, D15/2, D16/7, D17/2). CWE accuracy (D6, weight 12) is the single heaviest dimension in any profile, reflecting the researcher's need for precise vulnerability classification.

Head of Engineering. The engineering leader optimizes for developer experience and CI/CD integration. This lens selects 13 dimensions and emphasizes Detection (D2/5, D3/12, D7/8, D8/10 = 35) and

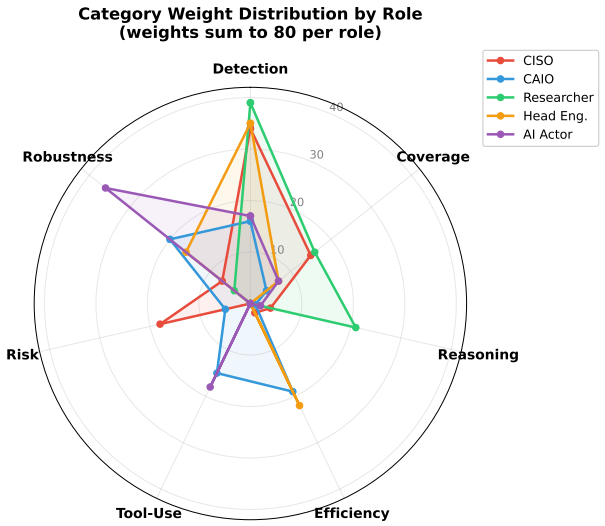

Figure 1: Category-level weight distribution for the five stakeholder roles. Each axis represents one of the 7 measurement categories (weights sum to 80 per role).

Efficiency (D18/7, D21/7, D22/5, D23/3 = 22). Precision (D3, weight 12) and Actionable Finding Rate (D8, weight 10) are the top-weighted dimensions because false positives erode developer trust and findings without location data are not actionable.

AI as Actor. This lens evaluates the model's fitness for autonomous operation. It selects 13 dimensions and places 36 of 80 weight points on Robustness (D31/3, D32/6, D33/6, D34/12, D35/9), 18 on Tool-Use (D25/5, D26/5, D27/8), and 17 on Detection (D1/10, D4/7). Autonomous Completion (D34, weight 12) and Graceful Degradation (D35, weight 9) are the distinctive dimensions: the agent must operate without crashing and maintain performance on both common and rare vulnerability classes.

Figure 1 visualizes the category-level weight distribution for each role. The five profiles form distinct shapes: the CISO and Researcher concentrate on Detection and Reasoning, the Head of Engineering balances Detection with Efficiency and Robustness, the CAIO distributes weight broadly, and the AI Actor concentrates heavily on Robustness.

3.2 Evaluation Layers

SecLens-R builds on the SecLens benchmark [Halder et al., 2026], which evaluates models in two layers:

- **Layer 1, Code-in-Prompt (CIP):** The vulnerable function is provided directly in the prompt. The model must identify whether the code is vulnerable, classify the CWE, and provide analysis from a single code snippet. This tests pure reasoning ability without tool use.
- **Layer 2, Tool-Use (TU):** The model is given access to a sandboxed repository clone and three tools: `read_file`, `search`, and `list_dir`. This tests real-world auditing ability, requiring multi-turn repository navigation.

Each task awards up to 3 points for true positive (vulnerable) tasks: 1 point for correct verdict, 1 for correct CWE identification, and 1 for correct location (file path + line range with IoU above threshold). Negative tasks (post-patch code) award 1 point for correct verdict.

The distinction between CIP and TU is central to the dimension design: several dimensions (D7, D8, D24–D27) are only computable in TU mode, and the scoring formula (Section 3.5) adjusts automatically when dimensions are unavailable. We use the abbreviations CIP and TU throughout the remainder of this paper.

Table 2: Seven measurement categories and their constituent dimensions.

ID	Category	Dimensions
A	Detection	D1–D8
B	Coverage & Consistency	D9–D13
C	Reasoning & Evidence	D14–D17
D	Operational Efficiency	D18–D23
E	Tool-Use & Navigation	D24–D27
F	Risk & Severity	D28–D30
G	Robustness	D31–D35

3.3 Dimension Taxonomy

All five roles draw from a shared pool of 35 dimensions organized into 7 categories. Table 2 presents the category taxonomy.

The shared-pool design means that when two roles both include a dimension (e.g., D1: MCC), they measure the same underlying quantity but assign different weights. This enables direct comparison of how weight allocation, rather than measurement definition, drives divergent evaluations.

3.4 Weight Assignment Methodology

Each role has a weight vector $\mathbf{w}^{(r)} = (w_1^{(r)}, w_2^{(r)}, \ldots)$ over a selected subset $S^{(r)} \subseteq \{D1, \ldots, D35\}$ with $|S^{(r)}| \in [12, 16]$. Weights satisfy:

$$\sum_{i \in S^{(r)}} w_i^{(r)} = 80 \quad \forall r \tag{1}$$

Weight assignment followed a structured process:

1. **Role profiling.** For each role, we identified the top decision criteria based on published job descriptions, industry frameworks (NIST CSF, ISO 27001, OWASP), and practitioner judgment.
2. **Dimension selection.** Each role selects 12–16 dimensions from the shared pool of 35.
3. **Weight distribution.** The 80 weight points are distributed among the selected dimensions, with higher weights for dimensions central to the role's decision question.
4. **Verification.** The final vector is verified to sum to exactly 80.

Table 3 presents the category-level weight distribution. The profiles are defined in YAML files, making them straightforward to customize for organizational needs.

Several patterns emerge. The CISO allocates the most weight to Detection (34) and Risk & Severity (18), reflecting the primacy of trustworthy, severity-aware threat detection. The CAIO balances Robustness (20), Efficiency (19), Detection (16), and Tool-Use (15), seeking models that are both capable and cost-effective. The Security Researcher concentrates on Detection (39) and Reasoning (21), valuing deep vulnerability understanding. The Head of Engineering distributes weight between Detection (35) and Efficiency (22), prioritizing actionable, fast, affordable scanning. The AI-as-Actor lens heavily weights Robustness (36) and Tool-Use (18), testing whether the model can operate autonomously without failure.

3.5 Composite Decision Score

For a model m evaluated under role r, let $s_i^{(m)}$ denote the normalized score (in $[0,1]$) on dimension i, and let $A^{(r)} \subseteq S^{(r)}$ denote the set of *available* dimensions (those for which data exists in the current evaluation). The composite Decision Score is:

$$D^{(r)}(m) = \frac{\sum_{i \in A^{(r)}} w_i^{(r)} \cdot s_i^{(m)}}{\sum_{i \in A^{(r)}} w_i^{(r)}} \times 100 \tag{2}$$

The denominator uses the sum of available weights rather than the fixed total of 80. This dynamic exclusion handles cases where certain dimensions cannot be computed:

- **CIP layer:** Tool-use dimensions D24–D27 and location dimensions D7, D8 are excluded (no tools, no file-level location in Layer 1).
- **No severity data:** Dimensions D28–D30 are excluded when task severity annotations are absent.
- **No SAST FP tasks:** Dimension D13 is excluded when the dataset contains no SAST false-positive tasks.

Normalization. We apply four strategies to map raw dimension values to $[0,1]$:

$$s_i = \begin{cases} \text{clamp}(v_i, 0, 1) & \text{if Ratio (D2–D17, D26–D35)} \\ \frac{v_i+1}{2} & \text{if MCC (D1 only)} \\ 1 - \min\left(\frac{v_i}{c_i}, 1\right) & \text{if Lower-is-better (D18,D19,D21,D23–D25)} \\ \min\left(\frac{v_i}{c_i}, 1\right) & \text{if Higher-is-better (D20,D22)} \end{cases} \tag{3}$$

where v_i is the raw value and c_i is a fixed reference cap. The caps are: D18 = \$0.50/task, D19 = \$2.00/TP, D20 = 100 MCC/\$, D21 = 120s, D22 = 60 tasks/min, D23 = 50K tokens, D24 = 30 tool calls, D25 = 20 turns. Fixed caps eliminate cohort-relative normalization artifacts: a model's score does not change when the evaluation cohort changes.

Grading Scale. Decision Scores map to letter grades: A $\geq$ 75, B $\geq$ 60, C $\geq$ 50, D $\geq$ 40, F $<$ 40.

4 Dimension Catalog

4.1 Master Dimension Table

Table 4 presents all 35 shared dimensions with their category, description, and normalization strategy. Descriptions are drawn from the computational definitions in the implementation.

4.2 Role Weight Profiles

Each role selects a subset of dimensions and assigns integer weights summing to 80. Tables 5–9 present the five profiles.

4.3 Dimension Availability by Layer

Not all 35 dimensions can be computed in every evaluation setting. In the CIP (Code-in-Prompt) layer, the model receives the vulnerable function directly in the prompt without access to tools or repository navigation. This means:

- **Tool-Use dimensions** (D24–D27) are excluded: no tool calls occur.
- **Location dimensions** (D7, D8) are excluded: CIP provides code inline, so file-level location is not evaluated.

Table 3: Category-level weight distribution across roles (weights sum to 80 per role).

Category	CISO	CAIO	Researcher	Head Eng.	AI Actor
A: Detection	34	16	39	35	17
B: Coverage & Consistency	15	4	16	7	7
C: Reasoning & Evidence	4	1	21	0	2
D: Operational Efficiency	2	19	0	22	0
E: Tool-Use & Navigation	0	15	0	0	18
F: Risk & Severity	18	5	0	0	0
G: Robustness	7	20	4	16	36
Total	**80**	**80**	**80**	**80**	**80**

- **SAST FP dimension** (D13) is excluded: the current dataset contains no SAST false-positive tasks.
- **Severity dimensions** (D28–D30) require task-level severity annotations; they are included when severity data is available.

When dimensions are excluded, Equation 2 adjusts the denominator so that the Decision Score reflects only the available evidence. A role that weights excluded dimensions heavily will have a smaller effective denominator, but the score still ranges from 0 to 100.

5 Evaluation Methodology

5.1 SecLens Integration

As described in Section 3.2, SecLens-R extends the SecLens benchmark [Halder et al., 2026], which evaluates models across CIP and TU layers with per-task scoring of verdict, CWE, and location. The role-specific dimensions consume the per-task result records produced by SecLens, including:

- **Scores:** verdict (0/1), CWE match (0/1), location match (0/1), total earned
- **Parse result:** status (FULL/PARTIAL/FAILED), parsed output fields (verdict, CWE, location, reasoning, evidence chain with source/sink/flow)
- **Metrics:** cost_usd, total_tokens, wall_time_s, tool_calls, turns
- **Task metadata:** task_type, task_category, task_language, task_severity

5.2 Scoring Pipeline

The end-to-end scoring pipeline proceeds as follows:

1. **Run evaluation.** Execute SecLens with a specified model, dataset, layer, and prompt preset. This produces a JSONL file of per-task `TaskResult` records.
2. **Compute dimensions.** For each of the 35 shared dimensions, compute the raw value from the result records using the dimension functions.
3. **Normalize.** Apply the dimension-specific normalization strategy (Ratio, MCC, Lower-is-better, or Higher-is-better) to map each raw value to $[0, 1]$.
4. **Select and weight.** For each role, select the role's dimension subset, multiply each normalized score by its role-specific weight, and compute the Decision Score via Equation 2.
5. **Grade and report.** Map the Decision Score to a letter grade and produce a per-role report with category subtotals and individual dimension scores.

6 Experimental Design

6.1 Models Under Evaluation

We evaluate 12 models spanning four providers:

Anthropic: Claude Opus 4.6, Claude Sonnet 4.6, Claude Haiku 4.5.

Google: Gemini 3.1 Pro Preview, Gemini 3 Flash Preview, Gemini 2.5 Pro, Gemini 2.5 Flash.

OpenAI: GPT-5.4.

Other (via OpenRouter): Qwen3-Coder, Qwen3-Coder-Plus, Kimi K2.5, Grok Code Fast 1.

This selection spans frontier reasoning models (Opus 4.6, Gemini 3.1 Pro), mid-tier models (Sonnet 4.6, Gemini 2.5 Pro), cost-optimized models (Haiku 4.5, Gemini 2.5 Flash), and open-weight models accessed through third-party routing (Qwen3, Kimi, Grok).

6.2 Dataset

We use the SecLens dataset, which contains tasks derived from confirmed CVEs across multiple CWE categories and programming languages. Table 10 summarizes the dataset statistics.

Table 4: Master dimension catalog (35 dimensions). Strategy: R = Ratio, M = MCC, L = Lower-is-better, H = Higher-is-better.

ID	Name	Category	Description	Norm.
D1	MCC	Detection	Matthews Correlation Coefficient [Matthews, 1975, Chicco and Jurman, 2020]; balanced metric robust to class imbalance	M
D2	Recall	Detection	True positive rate; fraction of vulnerable code correctly flagged	R
D3	Precision	Detection	Positive predictive value; fraction of "vulnerable" verdicts that are correct	R
D4	F1	Detection	Harmonic mean of precision and recall	R
D5	True Negative Rate	Detection	Specificity; fraction of non-vulnerable code correctly cleared	R
D6	CWE Accuracy	Detection	Correct CWE-ID among true positive detections	R
D7	Mean Location IoU	Detection	Average intersection-over-union of predicted vs. ground-truth line ranges (TU only)	R
D8	Actionable Finding Rate	Detection	Fraction of TPs with verdict + CWE + location (fully actionable)	R
D9	CWE Coverage Breadth	Coverage	Fraction of CWE categories with $\geq$1 correct detection	R
D10	Worst Category Floor	Coverage	Minimum F1 across all vulnerability categories; no blind spots	R
D11	Cross-Language Consistency	Coverage	$1 -$ StdDev of F1 across programming languages	R
D12	Worst Language Floor	Coverage	Minimum F1 across all programming languages	R
D13	SAST FP Filtering	Coverage	Accuracy on SAST false-positive tasks (currently excluded; no SAST FP tasks in dataset)	R
D14	Evidence Completeness	Reasoning	Fraction of TPs with source, sink, and data flow evidence	R
D15	Reasoning Presence	Reasoning	Fraction of all responses with reasoning field populated	R
D16	Reasoning + Correct Verdict	Reasoning	Fraction with reasoning AND correct verdict	R
D17	FP Reasoning Quality	Reasoning	Among false positive predictions, fraction with reasoning present	R
D18	Cost per Task	Efficiency	Average USD per task	L
D19	Cost per True Positive	Efficiency	Dollars per correctly detected vulnerability	L
D20	MCC per Dollar	Efficiency	MCC divided by total cost; quality per dollar	H
D21	Wall Time per Task	Efficiency	Average seconds per task	L
D22	Throughput	Efficiency	Tasks per minute	H
D23	Tokens per Task	Efficiency	Average total tokens consumed	L
D24	Tool Calls per Task	Tool-Use	Average number of tool invocations (TU only)	L
D25	Turns per Task	Tool-Use	Average conversation turns (TU only)	L
D26	Navigation Efficiency	Tool-Use	Fraction of tool calls that accessed relevant files (TU only)	R
D27	Tool Effectiveness	Tool-Use	Among tool-using tasks, fraction with score > 0 (TU only)	R
D28	Severity-Weighted Recall	Risk	Recall weighted by advisory-reported severity (critical 4×, high 3×, medium 2×, low 1×)	R
D29	Critical Miss Rate	Risk	1− miss rate on critical/high severity vulnerabilities	R
D30	Severity Coverage	Risk	Fraction of severity tiers with $\geq$1 correct detection	R
D31	Parse Success Rate	Robustness	Fraction of responses with parseable output (FULL or PARTIAL)	R
D32	Format Compliance	Robustness	FULL parse rate; fully schema-compliant structured output	R
D33	Error Rate	Robustness	1− fraction of tasks that crashed or produced errors	R
D34	Autonomous Completion	Robustness	Fraction of tasks completing without error or parse failure	R
D35	Graceful Degradation	Robustness	$1 - \lvert\text{common\_acc} - \text{rare\_acc}\rvert$; stable performance across common and rare CWEs	R

Table 5: CISO weight profile (16 dimensions, $\Sigma = 80$).

Dim	Name	Wt
D1	MCC	10
D2	Recall	8
D3	Precision	6
D5	True Negative Rate	2
D6	CWE Accuracy	5
D8	Actionable Finding Rate	5
D9	CWE Coverage Breadth	4
D10	Worst Category Floor	6
D11	Cross-Language Consistency	3
D14	Evidence Completeness	4
D18	Cost per Task	2
D28	Severity-Weighted Recall	10
D29	Critical Miss Rate	8
D33	Error Rate	3
D34	Autonomous Completion	3
D35	Graceful Degradation	1
	Total	**80**

Table 6: CAIO weight profile (14 dimensions, $\Sigma = 80$).

Dim	Name	Wt
D1	MCC	9
D4	F1	7
D9	CWE Coverage Breadth	4
D15	Reasoning Presence	1
D18	Cost per Task	5
D20	MCC per Dollar	8
D22	Throughput	6
D25	Turns per Task	5
D26	Navigation Efficiency	3
D27	Tool Effectiveness	7
D30	Severity Coverage	5
D31	Parse Success Rate	4
D32	Format Compliance	6
D34	Autonomous Completion	10
	Total	**80**

Table 7: Security Researcher weight profile (13 dimensions, $\Sigma = 80$).

Dim	Name	Wt
D1	MCC	8
D2	Recall	6
D6	CWE Accuracy	12
D7	Mean Location IoU	10
D8	Actionable Finding Rate	3
D9	CWE Coverage Breadth	7
D10	Worst Category Floor	5
D11	Cross-Language Consistency	4
D14	Evidence Completeness	10
D15	Reasoning Presence	2
D16	Reasoning + Correct Verdict	7
D17	FP Reasoning Quality	2
D35	Graceful Degradation	4
	Total	**80**

The dataset includes two task types. **True Positive (TP)** tasks contain pre-patch code with a confirmed vulnerability. **Post-Patch** tasks contain the same function after the official fix has been applied. Each TP task is paired with its post-patch counterpart, yielding a balanced 203/203 split.

Vulnerability categories. The 8 categories align with the OWASP Top 10:2021 taxonomy [OWASP Foundation, 2021], with 6 direct mappings and 2 extended categories. Table 11 shows the distribution.

Memory Safety (CWE-119 family, covering C/C++ buffer overflows) and Improper Input Validation (CWE-20 family) extend beyond the OWASP Top 10 to cover vulnerability classes not fully captured by the injection and SSRF categories.

Programming languages. 10 languages are represented: PHP (54 tasks), Go (54), Python (48), C# (46), Ruby (36), Java (36), C (36), Rust (34), JavaScript (32), and C++ (30).

Severity distribution. Among the 203 TP tasks with advisory-reported severity: Critical (25), High (74), Medium (83), Low (21).

6.3 Evaluation Protocol

Each model was evaluated under the following conditions:

- **Both layers:** CIP (Code-in-Prompt) and TU (Tool-Use) for all models supporting tool calling.
- **Prompt preset:** `base` (standard instructions).
- **Mode:** `guided` (category hint provided).
- **Full dataset:** All 406 tasks per run, seed=42.

For each model × layer combination, we compute all applicable dimensions and the five role-specific Decision Scores. This yields 23 total evaluation runs (not all models completed both layers).

7 Results and Analysis

7.1 Overall Leaderboard

Table 12 presents the full leaderboard across both CIP and TU layers. CIP scores range from 37.3% (Qwen3-Coder) to 49.6% (Gemini 3 Flash Preview). TU scores are consistently lower, ranging from 31.1% (GPT-5.4) to 45.8% (Gemini 3.1 Pro Preview), reflecting the added difficulty of multi-turn repository navigation.

Table 8: Head of Engineering weight profile (13 dimensions, $\Sigma = 80$).

Dim	Name	Wt
D2	Recall	5
D3	Precision	12
D5	True Negative Rate	4
D7	Mean Location IoU	8
D8	Actionable Finding Rate	10
D12	Worst Language Floor	3
D18	Cost per Task	7
D21	Wall Time per Task	7
D22	Throughput	5
D23	Tokens per Task	3
D31	Parse Success Rate	7
D32	Format Compliance	3
D33	Error Rate	6
	Total	**80**

Table 9: AI-as-Actor weight profile (13 dimensions, $\Sigma = 80$).

Dim	Name	Wt
D1	MCC	10
D4	F1	7
D9	CWE Coverage Breadth	3
D11	Cross-Language Consistency	4
D14	Evidence Completeness	2
D25	Turns per Task	5
D26	Navigation Efficiency	5
D27	Tool Effectiveness	8
D31	Parse Success Rate	3
D32	Format Compliance	6
D33	Error Rate	6
D34	Autonomous Completion	12
D35	Graceful Degradation	9
	Total	**80**

Cost data was tracked for Anthropic, Google, and OpenAI models. The four OpenRouter-routed models (Kimi K2.5, Grok Code Fast 1, Qwen3-Coder, Qwen3-Coder-Plus) did not have cost tracking available through the provider, so cost-derived dimensions (D18, D19, D20) are excluded from their scoring and marked N/T in cost columns.

7.2 Per-Role Decision Scores (CIP Layer)

Table 13 presents the Decision Scores across all 12 models and 5 roles for the CIP layer. Each cell shows the letter grade and numeric score. Models are sorted by leaderboard score.

Table 10: Dataset statistics.

Attribute	Count
Total tasks	406
True Positive tasks	203
Post-Patch tasks	203
Source projects	93
OWASP categories	8
Programming languages	10
Distinct CVEs	∼203

Table 11: Vulnerability category distribution (TP + Post-Patch).

Category	OWASP	Tasks
Broken Access Control	A01:2021	82
Cryptographic Failures	A02:2021	64
Injection	A03:2021	62
Improper Input Validation	Extended	58
SSRF	A10:2021	46
Auth Failures	A07:2021	38
Data Integrity Failures	A08:2021	36
Memory Safety	Extended	20

Several patterns emerge from the data:

AI Actor is universally lenient. All 12 models earn an A grade (77.9–87.5). This role places 36 of 80 weight points on Robustness, and most frontier models parse and comply well. The narrow score range (9.6 points from worst to best) indicates that current models have largely solved the robustness challenges this lens measures.

CISO is the strictest lens. Scores range from D (45.2, Qwen3-Coder) to B (73.3, Gemini 3 Flash). This role weights severity-weighted recall (D28, weight 10) and critical miss rate (D29, weight 8), which sharply penalize models that fail to detect high-severity vulnerabilities. Only 8 of 12 models achieve B or higher.

Head of Engineering favors different models. GPT-5.4 (A, 76.7), Qwen3-Coder (A, 76.3), and Qwen3-Coder-Plus (A, 76.9) earn their highest grades under this lens, while scoring C or D for the CISO. This role rewards high precision (D3, weight 12), actionable findings (D8, weight 10), fast wall times (D21, weight 7), and low cost (D18, weight 7). Models with conservative prediction strategies (high precision, lower recall) perform well here.

Leaderboard rank does not predict role rank. Gemini 3 Flash leads both the leaderboard (49.6%) and the CISO lens (73.3). But Claude Haiku 4.5 ranks only 8th on the leaderboard (43.8%) yet scores 2nd for the CISO (71.2), a jump of 6 positions. Conversely,

Table 12: Full leaderboard across CIP and TU layers, sorted by CIP score. Cost shown where tracked; N/T = not tracked.

Model	CIP Score (%)	CIP Cost	TU Score (%)	TU Cost
Gemini 3 Flash Preview	49.6	$15.87	44.2	$184.21
Gemini 3.1 Pro Preview	48.2	$44.81	45.8	$639.09
Claude Sonnet 4.6	47.6	$6.76	42.1	$359.76
Kimi K2.5	46.8	N/T	–	–
Gemini 2.5 Pro	46.2	$13.45	35.2	$57.91
Grok Code Fast 1	44.1	N/T	34.4	N/T
Gemini 2.5 Flash	44.3	$3.23	34.2	$4.46
Claude Haiku 4.5	43.8	$2.11	37.8	$104.06
Claude Opus 4.6	41.7	$7.56	39.0	$371.30
Qwen3-Coder-Plus	41.2	N/T	34.9	N/T
GPT-5.4	39.9	$2.97	31.1	$23.16
Qwen3-Coder	37.3	N/T	32.5	N/T

Table 13: Per-role Decision Scores (CIP layer). Grade thresholds: A $\geq$ 75, B $\geq$ 60, C $\geq$ 50, D $\geq$ 40, F $<$ 40. Sorted by leaderboard score (LB).

Model	LB %	CISO	CAIO	Researcher	Head Eng.	AI Actor
Gemini 3 Flash Preview	49.6	B (73.3)	B (68.1)	B (71.0)	B (66.2)	A (87.5)
Gemini 3.1 Pro Preview	48.2	B (67.5)	B (67.0)	B (65.7)	B (63.8)	A (85.7)
Claude Sonnet 4.6	47.6	B (65.7)	B (68.4)	B (64.2)	B (73.9)	A (85.6)
Kimi K2.5	46.8	B (68.0)	B (67.8)	B (67.0)	B (65.1)	A (86.4)
Gemini 2.5 Pro	46.2	B (66.2)	B (67.9)	B (65.2)	B (71.3)	A (86.3)
Grok Code Fast 1	44.1	C (58.7)	B (67.5)	B (60.2)	B (73.0)	A (83.8)
Gemini 2.5 Flash	44.3	B (61.3)	B (67.9)	B (61.1)	B (72.3)	A (84.9)
Claude Haiku 4.5	43.8	B (71.2)	B (69.1)	B (68.2)	B (73.3)	A (85.9)
Claude Opus 4.6	41.7	C (51.0)	B (65.6)	C (55.6)	B (72.9)	A (80.2)
Qwen3-Coder-Plus	41.2	C (51.1)	B (68.0)	C (54.2)	A (76.9)	A (81.2)
GPT-5.4	39.9	D (48.4)	B (67.0)	C (54.1)	A (76.7)	A (79.2)
Qwen3-Coder	37.3	D (45.2)	B (64.0)	C (52.9)	A (76.3)	A (77.9)

GPT-5.4 ranks 11th on the leaderboard but 2nd for Head of Engineering.

CAIO scores are remarkably stable. All 12 models score B (64.0–69.1), the tightest band of any role. The 5.1-point spread suggests that current models have comparable profiles on the balanced mix of efficiency, capability, and robustness dimensions the CAIO weights.

Figure 2 visualizes these patterns. The AI Actor bars (purple) form a consistently high ceiling, while the CISO bars (red) show the widest variation. The Head of Engineering bars (yellow) rise on the right side of the chart where conservative models (GPT-5.4, Qwen3) appear, crossing over the CISO bars.

7.3 Per-Category Performance

Table 14 presents the top-3 and worst models by F1 for each vulnerability category in the CIP layer.

No single model dominates all categories. Six different models lead at least one category. Gemini 3 Flash leads Cryptographic Failures and Data Integrity; Haiku 4.5 leads Memory Safety and Input Validation; Kimi K2.5 leads Broken Access Control and Auth Failures; Gemini 3.1 Pro leads Injection; and Sonnet 4.6 leads SSRF.

SSRF is where Claude models excel. Sonnet 4.6 (0.690) and Opus 4.6 (0.689) rank #1 and #2 for Server-Side Request Forgery, while performing worse on other categories. This suggests architecture-specific strengths in reasoning about network request patterns.

Authentication Failures is the hardest category. The best F1 is only 0.585 (Kimi K2.5), and Opus 4.6 scores 0.000 (a complete miss on all authentication-related vulnerabilities in CIP mode). This category likely requires multi-file reasoning about authentication flows that is difficult from a single code snippet.

Qwen3-Coder is consistently worst. It ranks

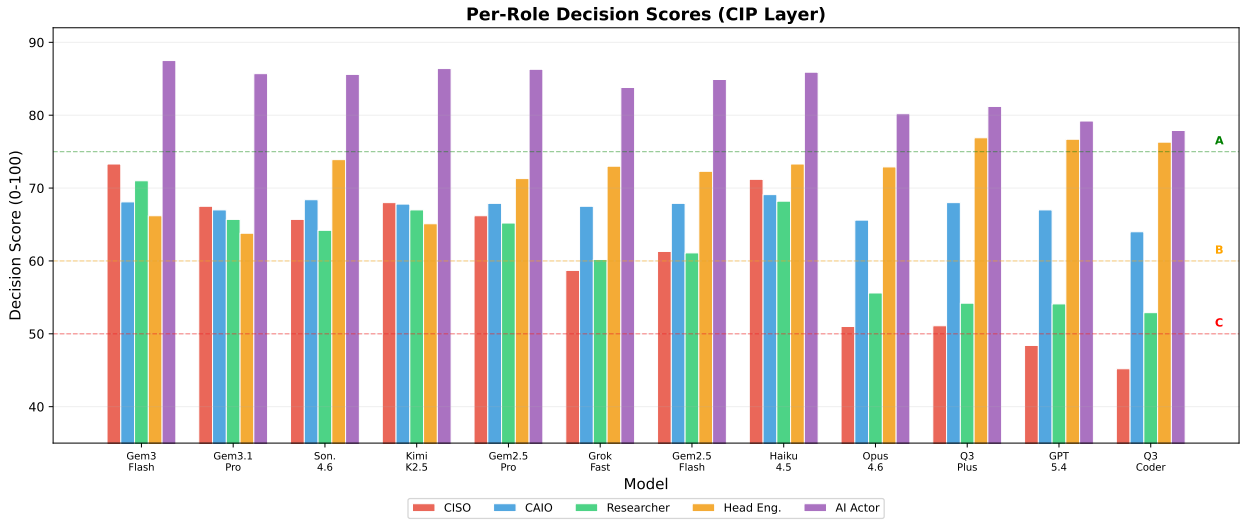

Figure 2: Per-role Decision Scores for 12 models (CIP layer). Dashed lines mark grade thresholds: A $\geq$ 75, B $\geq$ 60, C $\geq$ 50. Models sorted by leaderboard score (left to right).

Table 14: Per-category F1 leaders and worst performers (CIP layer).

Category	**#1 (F1)**	**#2 (F1)**	**#3 (F1)**	**Worst (F1)**
Broken Access Ctrl.	Kimi K2.5 (0.667)	Gemini 3.1 Pro (0.660)	Gemini 3 Flash (0.647)	Qwen3-Coder (0.128)
Cryptographic Failures	Gemini 3 Flash (0.676)	Gemini 3.1 Pro (0.667)	Haiku 4.5 (0.633)	Qwen3-Coder (0.118)
Auth Failures	Kimi K2.5 (0.585)	Gemini 2.5 Pro (0.579)	Gemini 3.1 Pro (0.578)	Opus 4.6 (0.000)
Input Validation	Haiku 4.5 (0.675)	Kimi K2.5 (0.649)	Gemini 2.5 Flash (0.638)	Qwen3-Coder (0.125)
Injection	Gemini 3.1 Pro (0.632)	Gemini 3 Flash (0.618)	Gemini 2.5 Flash (0.603)	Qwen3-Coder (0.062)
Memory Safety	Haiku 4.5 (0.690)	Gemini 3 Flash (0.667)	Gemini 2.5 Flash (0.640)	Qwen3-Coder (0.308)
SSRF	Sonnet 4.6 (0.690)	Opus 4.6 (0.689)	Qwen3-Plus (0.682)	Qwen3-Coder (0.512)
Data Integrity	Gemini 3 Flash (0.680)	Haiku 4.5 (0.679)	Gemini 2.5 Flash (0.625)	Qwen3-Coder (0.200)

last in 7 of 8 categories. Inspection of its outputs reveals very high precision but near-zero recall: the model rarely predicts "vulnerable," so it misses almost all true positives while avoiding false positives.

Haiku 4.5 punches above its weight. Despite ranking 8th on the overall leaderboard, it leads Memory Safety (0.690) and Input Validation (0.675). The CISO lens captures this: Haiku ranks 2nd (71.2) under the CISO lens because these category strengths align with the CISO's severity-weighted recall priorities.

Per-language variation. Performance also varies by language. For C code, Gemini 3.1 Pro leads (F1 = 0.750) while Qwen3-Coder trails (0.100). For JavaScript, Gemini 3.1 Pro again leads (0.684) while Qwen3-Coder scores 0.000. For Go, Haiku 4.5 leads (0.647) while GPT-5.4 trails (0.278).

Table 15 presents the complete model-by-category F1 matrix, enabling readers to identify which model is strongest for their specific vulnerability categories of concern.

Figure 3 presents the complete F1 matrix as a heatmap, where green cells indicate strong performance and red cells indicate weakness. The diagonal pattern of strengths across models is visible: no single column is uniformly green, confirming that category-specific evaluation is necessary.

Category performance predicts role divergence. Models with high recall across categories (Gemini 3 Flash, Haiku 4.5) score well for the CISO, whose D28 (Severity-Weighted Recall) and D29 (Critical Miss Rate) aggregate category-level detection. Models with concentrated recall in a few categories but high precision overall (GPT-5.4, Qwen3-Coder) score well for the Head of Engineering, whose D3 (Precision) dominates at weight 12. This table reveals why the same model can score 31 points apart under different lenses: the CISO lens penalizes category-level gaps visible in the Auth Failures and Injection columns, while the Engineering lens rewards the global precision visible in models that rarely flag code as vulnerable.

Table 15: F1 score by model and vulnerability category (CIP layer). Bold indicates best-in-category. Italics indicates worst. Models sorted by average F1.

Model	BAC	Crypto	Auth	Input	Inj.	Mem.	SSRF	D.Int.	Avg
Gemini 3 Flash	0.647	**0.676**	0.444	0.627	0.618	0.667	0.667	**0.680**	0.628
Claude Haiku 4.5	0.586	0.633	0.545	**0.675**	0.507	**0.690**	0.667	0.679	0.623
Gemini 3.1 Pro	0.660	0.667	0.578	0.618	**0.632**	0.583	0.600	0.500	0.605
Kimi K2.5	**0.667**	0.567	**0.585**	0.649	0.517	0.609	0.644	0.529	0.596
Gemini 2.5 Pro	0.635	0.562	0.579	0.627	0.585	0.526	0.643	0.514	0.584
Claude Sonnet 4.6	0.590	0.625	0.514	0.571	0.518	0.526	**0.690**	0.537	0.571
Gemini 2.5 Flash	0.457	0.500	0.191	0.638	0.603	0.640	0.623	0.625	0.535
Grok Code Fast 1	0.530	0.367	0.529	0.561	0.440	0.588	0.667	0.414	0.512
Claude Opus 4.6	0.231	0.491	*0.000*	0.408	0.263	0.476	0.689	0.333	0.361
Qwen3-Plus	0.310	0.216	0.320	0.417	0.350	0.400	0.682	0.320	0.377
GPT-5.4	0.182	0.222	0.333	0.256	0.171	0.353	0.609	0.214	0.293
Qwen3-Coder	*0.128*	*0.118*	0.100	*0.125*	*0.062*	*0.308*	*0.512*	*0.200*	*0.194*

7.4 Category-Role Interaction

The relationship between per-category performance and role scores explains many of the divergences in Table 13. We highlight three instructive cases:

Case 1: Claude Haiku 4.5 (CISO 2nd, Leaderboard 8th). Haiku has the highest recall for Memory Safety (0.909), Input Validation (0.966), and competitive recall across Auth Failures (0.923) and SSRF (0.800). The CISO lens rewards this broad, high-recall coverage through D2 (Recall, weight 8) and D28 (Severity-Weighted Recall, weight 10). Despite lower precision (which the Engineering lens penalizes), Haiku's category-level coverage makes it a strong CISO pick.

Case 2: GPT-5.4 (Eng. 2nd, CISO 11th). GPT-5.4 achieves high precision (0.800 on Broken Access Control, 0.750 on Injection) but very low recall (0.103 and 0.097 respectively). The Engineering lens rewards this through D3 (Precision, weight 12) and D33 (Error Rate, weight 6). The CISO lens penalizes it through D28 (Severity-Weighted Recall), which drops because GPT-5.4 misses the majority of true vulnerabilities including critical ones.

Case 3: Opus 4.6 (Auth Failures blind spot). Opus 4.6 achieves F1 = 0.000 on Authentication Failures (recall 0.000) while performing well on SSRF (F1 = 0.689). The CISO lens captures this via D10 (Worst Category Floor, weight 6), which drives the CISO score down to C (51.0). The AI Actor lens is unaffected because it does not include D10, illustrating how dimension selection creates lens-specific sensitivity to category gaps.

7.5 Cross-Role Divergence Analysis

We define the *Role Divergence Index* (RDI) for a model m:

Table 16: Role Divergence Index (CIP layer). Highest and lowest scoring roles shown.

Model	RDI	Best Role	Worst Role
Qwen3-Coder	31.1	AI Actor (77.9)	CISO (45.2)
GPT-5.4	30.8	AI Actor (79.2)	CISO (48.4)
Qwen3-Plus	30.1	AI Actor (81.2)	CISO (51.1)
Opus 4.6	29.2	AI Actor (80.2)	CISO (51.0)
Grok Code Fast	25.1	AI Actor (83.8)	CISO (58.7)
Gemini 2.5 Flash	23.6	AI Actor (84.9)	Researcher (61.1)
Sonnet 4.6	19.9	AI Actor (85.6)	Researcher (64.2)
Gemini 3.1 Pro	21.9	AI Actor (85.7)	Eng. (63.8)
Gemini 2.5 Pro	20.1	AI Actor (86.3)	Researcher (65.2)
Kimi K2.5	21.3	AI Actor (86.4)	Eng. (65.1)
Haiku 4.5	16.8	AI Actor (85.9)	CAIO (69.1)
Gemini 3 Flash	21.3	AI Actor (87.5)	Eng. (66.2)

$$\mathrm{RDI}(m) = \max_r D^{(r)}(m) - \min_r D^{(r)}(m) \qquad (4)$$

A high RDI indicates that the model's perceived value depends heavily on the stakeholder perspective. Table 16 presents the RDI for each model.

Qwen3-Coder has the highest RDI at 31.1 points: it earns A for Head of Engineering (76.3) and AI Actor (77.9) but D for CISO (45.2). GPT-5.4 shows a similar pattern with 30.8 points of divergence. These models have conservative prediction strategies (high precision, low recall) that satisfy engineering and autonomy lenses but fail the CISO's severity-weighted recall requirements.

Haiku 4.5 has the lowest RDI at 16.8 points, indicating the most balanced profile across stakeholder perspectives. Its strong performance on both detection accuracy and operational dimensions produces consistent grades (B across all non-AI-Actor roles).

The worst-scoring role is nearly always the CISO (for 7 of 12 models), while the best-scoring role is

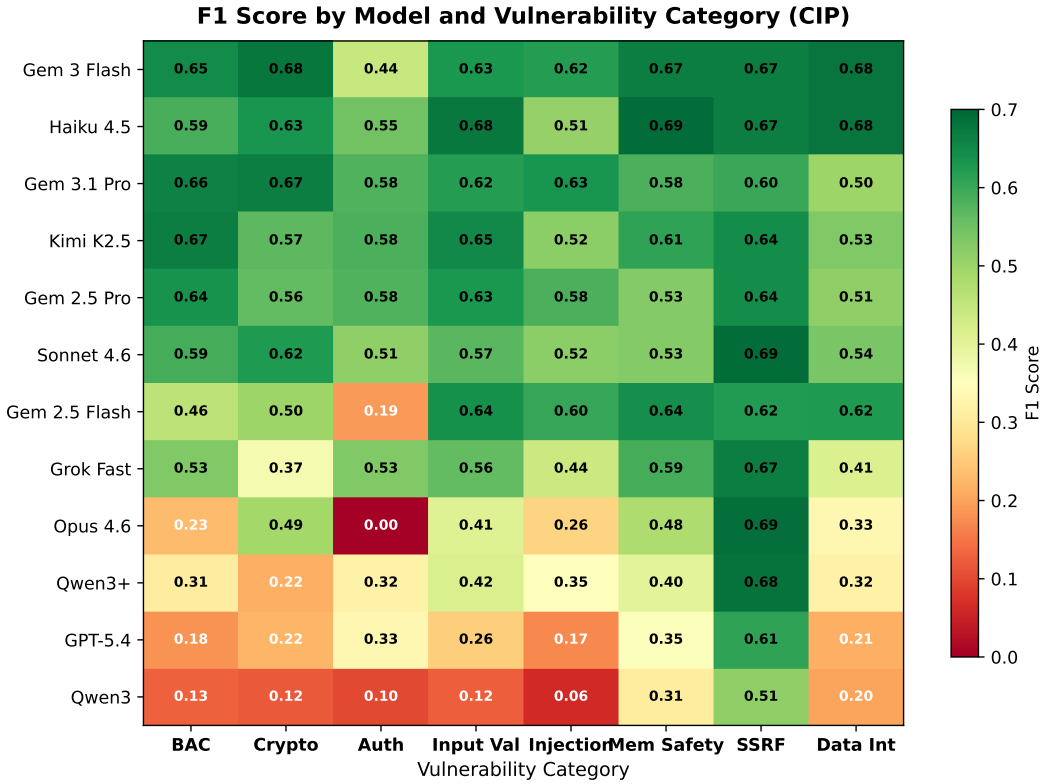

Figure 3: F1 score heatmap by model and vulnerability category (CIP layer). Models sorted by average F1. Green = strong, red = weak.

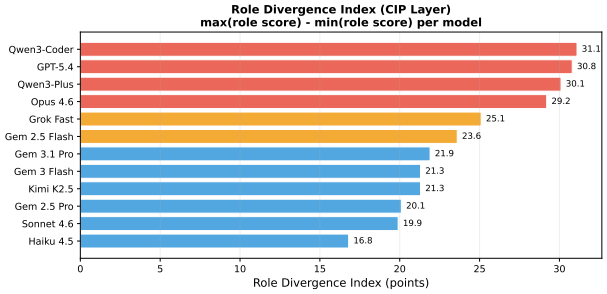

Figure 4: Role Divergence Index per model (CIP layer). Red = high divergence ($\geq$28), yellow = moderate ($\geq$22), blue = low. Higher RDI means the model's value depends more on stakeholder perspective.

always the AI Actor. This structural asymmetry reflects the current state of frontier models: parse reliability and format compliance (which the AI Actor lens weights heavily) are largely solved, while severity-aware vulnerability detection (which the CISO lens weights heavily) remains challenging. Figure 4 visualizes the RDI distribution.

7.6 CIP vs. Tool-Use Layer Analysis

We evaluated models on both CIP and TU layers. Tool-Use runs show several consistent patterns:

- **Lower leaderboard scores.** The top CIP score is 49.6% (Gemini 3 Flash) vs. 45.8% (Gemini 3.1 Pro) for TU. Tool-use adds complexity: models must navigate repositories, choose which files to read, and synthesize information across multiple turns.
- **Higher cost.** TU runs are 10–100$\times$ more expensive than CIP. Gemini 3.1 Pro costs \$0.111/task in CIP but \$1.578/task in TU. Claude Haiku 4.5 costs \$0.005/task in CIP but \$0.256/task in TU.
- **Additional dimensions available.** TU runs enable location accuracy (D7, D8) and tool-use dimensions (D24–D27), providing a richer signal for roles like Security Researcher and AI-as-Actor.

Table 17 presents the TU layer role scores for all 11 models evaluated on tool-use. Comparing against the CIP scores in Table 13 reveals that TU scores are generally lower for the CISO and Researcher but more stable for AI Actor. Gemini 3.1 Pro Preview achieves the highest TU CISO score (B, 69.4), surpassing its

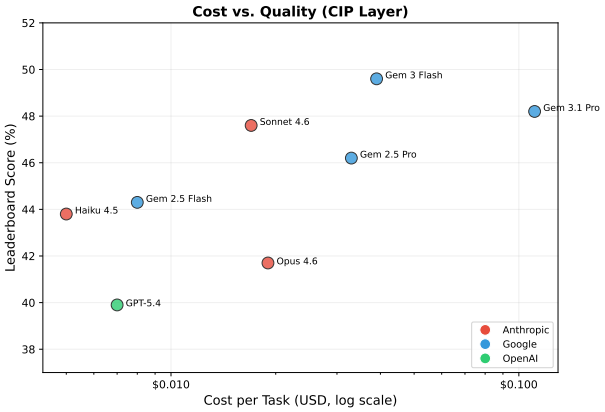

Figure 5: Cost per task vs. leaderboard score (CIP layer, 8 models with cost tracking). Log-scale x-axis. Higher and further left is better.

CIP CISO score (B, 67.5), one of the few models where TU improves role-specific performance.

The CIP layer provides a cost-effective baseline evaluation; the TU layer adds depth at a substantial cost premium. For cost-sensitive deployments, CIP evaluation alone may be sufficient, while organizations investing in agent-based security tools should evaluate on TU to capture tool-use quality. The TU layer reveals a key difference from CIP: the Head of Engineering scores drop sharply (most models score D), driven by high cost per task (D18) and slow wall times (D21) inherent to multi-turn tool-use evaluation.

7.7 Cost and Efficiency Analysis

Cost was tracked for 8 of 12 models (Anthropic, Google, and OpenAI). Table 18 presents the CIP cost data.

Figure 5 plots cost per task against leaderboard score for the 8 cost-tracked models, revealing diminishing returns at higher price points.

GPT-5.4 delivers the best quality-per-dollar. At $0.007/task, GPT-5.4 achieves an MCC/$ ratio of 0.042, the highest among tracked models. Its low cost makes it attractive for the Head of Engineering lens, which weights D18 (Cost per Task) and D20 (MCC per Dollar).

Cost does not predict accuracy. Gemini 3.1 Pro costs 15× more than GPT-5.4 per task but scores only 8.3 leaderboard points higher (48.2% vs. 39.9%). Gemini 3 Flash Preview, the top performer, costs 5.3× more than GPT-5.4.

Throughput varies widely. GPT-5.4 processes 15.1 tasks per minute while Gemini 3.1 Pro manages only 0.7 TPM, a 21× gap. For CI/CD integration where latency matters, this difference is significant. The Head of Engineering lens captures this via D22 (Throughput, weight 5).

Haiku 4.5 is the cheapest option. At $0.005/task ($2.11 total for 406 tasks), Haiku is the most economical model. Projected to 10K tasks/year, this amounts to roughly $52. Despite its low cost, Haiku scores B (71.2) for the CISO, outranking models that cost 3–20× more. This makes Haiku a strong candidate for organizations running frequent security scans on a budget.

TU layer costs are substantially higher. Gemini 2.5 Flash, the cheapest TU option at $0.011/task, still costs 37% more than its CIP equivalent. At the high end, Gemini 3.1 Pro costs $1.578/task in TU ($639.09 total), a 14× increase over its CIP cost. Token consumption drives this: TU runs for Haiku 4.5 average 236K tokens/task vs. roughly 2.8K in CIP, an 84× increase due to multi-turn repository navigation.

Cost tracking limitation. The four OpenRouter-routed models (Kimi K2.5, Grok Code Fast 1, Qwen3-Coder, Qwen3-Coder-Plus) did not report cost data. Their cost-derived dimensions (D18, D19, D20) are excluded via the dynamic denominator adjustment in Equation 2. This means their CAIO and Head of Engineering scores reflect fewer dimensions than models with cost tracking, which should be considered when comparing across providers.

7.8 Dimension Sensitivity Analysis

To identify which dimensions drive model differentiation for each role, we compute the Impact score:

$$\text{Impact}_i^{(r)} = w_i^{(r)} \cdot \text{Var}_m[s_i^{(m)}] \tag{5}$$

This product of weight and cross-model variance identifies dimensions that both matter to the role (high weight) and differentiate between models (high variance).

For the **CISO**, the highest-impact dimensions are D28 (Severity-Weighted Recall, weight 10) and D29 (Critical Miss Rate, weight 8), which have high cross-model variance because some models detect most critical vulnerabilities while others miss them entirely. D1 (MCC, weight 10) also contributes high impact.

For the **Head of Engineering**, D3 (Precision, weight 12) and D8 (Actionable Finding Rate, weight 10) drive differentiation. Models with conservative prediction strategies (Qwen3-Coder, GPT-5.4) achieve near-perfect precision, while models with higher recall but lower precision (Gemini 3 Flash) score lower on these dimensions but higher on D2 (Recall, weight 5).

For the **AI Actor**, variance is low across all Robustness dimensions (most models parse well), so the

Table 17: Per-role Decision Scores (TU layer). Grade thresholds: A $\geq$ 75, B $\geq$ 60, C $\geq$ 50, D $\geq$ 40, F $<$ 40. Sorted by TU leaderboard score.

Model	LB %	CISO	CAIO	Researcher	Head Eng.	AI Actor
Gemini 3.1 Pro Preview	45.8	B (69.4)	C (56.1)	B (65.0)	D (44.5)	A (75.4)
Gemini 3 Flash Preview	44.2	B (68.9)	C (55.9)	B (64.1)	D (44.5)	B (73.7)
Claude Sonnet 4.6	42.1	B (66.3)	C (56.1)	B (60.6)	D (43.9)	B (74.7)
Claude Opus 4.6	39.0	C (56.0)	C (55.6)	C (53.9)	D (43.8)	B (71.7)
Claude Haiku 4.5	37.8	B (66.5)	C (57.1)	B (60.3)	D (47.8)	B (72.8)
Gemini 2.5 Pro	35.2	C (59.7)	B (64.4)	C (55.2)	C (52.0)	A (79.7)
Qwen3-Coder-Plus	34.9	C (56.4)	B (63.7)	C (50.9)	C (54.1)	A (76.7)
Grok Code Fast 1	34.4	C (55.5)	B (64.3)	C (50.9)	C (50.6)	A (76.6)
Gemini 2.5 Flash	34.2	C (56.0)	B (67.4)	C (52.4)	C (56.3)	A (81.0)
Qwen3-Coder	32.5	D (48.9)	C (59.7)	D (48.2)	C (53.2)	B (70.8)
GPT-5.4	31.1	D (45.2)	B (64.2)	D (48.6)	B (60.8)	A (75.6)

Table 18: CIP layer cost analysis (8 models with cost tracking).

Model	$/task	Total	MCC/$	TPM
Haiku 4.5	$0.005	$2.11	0.019	9.8
GPT-5.4	$0.007	$2.97	0.042	15.1
Gemini 2.5 Flash	$0.008	$3.23	0.031	4.1
Sonnet 4.6	$0.017	$6.76	0.025	4.4
Opus 4.6	$0.019	$7.56	0.009	7.3
Gemini 2.5 Pro	$0.033	$13.45	0.011	2.2
Gemini 3 Flash	$0.039	$15.87	0.009	0.9
Gemini 3.1 Pro	$0.111	$44.81	0.004	0.7

Detection dimensions D1 (MCC, weight 10) and D4 (F1, weight 7) drive the modest differentiation that exists within the A-grade band.

8 Discussion

8.1 Limitations

Weight subjectivity. The weight vectors, while informed by domain expertise and industry frameworks, involve judgment. Different organizations may prioritize differently within the same role. We address this by storing profiles in YAML files that organizations can customize. Adding or modifying a role requires only a new YAML file specifying the dimension subset and weights.

Single-run evaluation. Without multi-run data, we cannot estimate confidence intervals on dimension scores. Dimensions computed from small subsets (e.g., Memory Safety with only 20 tasks, or rare CWE categories) may have high variance not captured in a single run.

No SAST FP tasks. The current dataset contains no SAST false-positive tasks, so D13 (SAST FP Filtering) is always excluded. This dimension would be valuable for evaluating a model's ability to distinguish tool-reported false alarms from real vulnerabilities.

Dataset size for rare categories. While 406 tasks is substantial, some categories have few tasks (Memory Safety: 20, Data Integrity: 36). Per-category metrics for these smaller groups carry wider implicit confidence intervals.

Paired task design. The balanced TP/post-patch design means that True Negative Rate (D5) and related specificity measures may be inflated relative to real-world scanning, where the ratio of vulnerable to non-vulnerable code is far more skewed.

Cost tracking gaps. Cost data was unavailable for 4 of 12 models (those routed via OpenRouter). This limits cost-efficiency analysis to 8 models and means that cost-derived dimensions are excluded for the remaining 4, reducing the effective dimensionality of their role scores.

Severity data on TP tasks only. Severity annotations (Critical, High, Medium, Low) exist only for true positive tasks. Post-patch tasks carry no severity label, so severity-weighted dimensions (D28–D30) are computed over the TP subset only.

8.2 Implications for Model Selection

Our results carry practical implications for organizations evaluating LLMs for security use:

No universal "best model" exists. Gemini 3 Flash Preview leads the aggregate leaderboard (49.6%), but Claude Haiku 4.5 is a better choice for a CISO (71.2 vs. 73.3) despite ranking 6 places lower overall. GPT-5.4, ranked 11th on the leaderboard, is the top choice for engineering teams (A, 76.7) and delivers the best cost-quality ratio ($0.007/task, MCC/$ = 0.042). Model selection must be anchored to the decision-maker's priorities.

Conservative models suit engineering; aggressive models suit security. Models with high precision and low recall (Qwen3-Coder, GPT-5.4) excel for the Head of Engineering, who values actionable findings and low false-positive rates. Models with high recall and broader detection (Gemini 3 Flash, Haiku 4.5) excel for the CISO, who values coverage and severity-aware detection. The same behavioral trait, conservative prediction, is a strength for one stakeholder and a weakness for another.

CIP evaluation is sufficient for most decisions. The CIP layer captures 65–70% of the framework's discriminative power at 10–100× lower cost than TU. For organizations primarily concerned with detection quality, reasoning, and cost, CIP-based role scores provide actionable guidance. The TU layer adds value mainly for AI-as-Actor and Security Researcher evaluations, where tool-use and location dimensions matter.

Category-specific weaknesses drive CISO failures. The CISO lens penalizes blind spots. Opus 4.6 scores 0.000 F1 on Authentication Failures; Qwen3-Coder achieves F1 below 0.13 on 4 of 8 categories. These category gaps, invisible in aggregate scores, determine whether a model can be trusted in a security program.

8.3 Comparison with Concurrent Work

Several benchmarks published concurrently address related but distinct problems. SecVulEval [Lu et al., 2025] is the largest single-language benchmark (25K C/C++ samples), but its scope is limited to C/C++ and it produces a single F1 score. SEC-bench [Lee et al., 2025] evaluates LLM agents on exploit generation and patching, tasks complementary to the detection focus of SecLens-R. TOSSS [Damie et al., 2026] frames detection as binary snippet selection, whereas SecLens requires CWE classification, location, and evidence generation, enabling richer dimension computation.

Our contribution is orthogonal: we do not propose a new detection benchmark, but a scoring layer that transforms any detection benchmark's per-task results into stakeholder-specific evaluations. The SecLens-R dimensions could be computed over SecVulEval or TOSSS results with minimal adaptation, provided those benchmarks emit per-task verdict, CWE, and location fields.

8.4 Future Work

Custom organizational profiles. The YAML-based architecture makes it straightforward to define custom roles. An interactive profile editor could help security teams create weight vectors aligned with their specific risk appetite and operational constraints.

SAST FP task collection. Curating a set of SAST-flagged but manually-confirmed non-vulnerable code samples would enable D13 and provide a direct measure of a model's ability to filter static analysis noise.

Multi-run evaluation. Evaluating each model multiple times with different seeds would yield confidence intervals on dimension scores and enable a cross-run stability dimension.

Expanded model coverage. Testing additional open-weight models (e.g., DeepSeek, Llama) and specialized security models would broaden the analysis. Running open-weight models locally would also resolve the cost tracking gap for OpenRouter-routed evaluations.

Temporal tracking. Evaluating successive model versions (e.g., Claude Sonnet 4.5 → 4.6) on the same dataset would enable regression risk assessment and capability progress tracking across role lenses.

Cross-benchmark integration. Applying the SecLens-R scoring layer to results from other benchmarks (SecVulEval [Lu et al., 2025], SEC-bench [Lee et al., 2025], TOSSS [Damie et al., 2026]) would test whether role-specific divergence generalizes beyond the SecLens dataset and task format.

9 Conclusion

We have presented SecLens-R, a multi-stakeholder evaluation framework for LLM-based security vulnerability detection. The framework defines 35 shared evaluation dimensions across 7 categories, applied through 5 role-specific weight profiles where each role selects 12–16 dimensions with weights summing to 80.

We evaluated 12 frontier models on 406 tasks spanning 93 projects, 10 programming languages, and 8 OWASP-aligned vulnerability categories. The results demonstrate that single-score benchmarks obscure information that matters for organizational decisions:

- Decision Scores diverge by up to 31 points across roles for the same model. Qwen3-Coder earns A (76.3) for Head of Engineering but D (45.2) for CISO.
- GPT-5.4 earns A for Head of Engineering (76.7) but D for CISO (48.4), driven by its conservative prediction strategy (high precision, low recall on critical vulnerabilities).
- Leaderboard rank does not predict role-specific rank. Claude Haiku 4.5, ranked 8th overall, scores 2nd for CISO (71.2).

- No single model dominates all vulnerability categories. Six different models lead at least one of the 8 OWASP-aligned categories.
- The AI Actor lens shows all models earning A grades, while the CISO lens shows grades ranging from D to B, confirming that model robustness is largely solved but severity-aware detection remains challenging.

The YAML-based weight profiles enable organizational customization: teams can adjust or create roles to match their specific risk appetites and operational priorities. The framework integrates directly with the SecLens evaluation pipeline, consuming existing per-task result records without requiring any modifications to the underlying benchmark.

We release the full implementation as open-source software to support organizational adoption and community extension.

Acknowledgments

The authors thank the open-source LLM and security research communities for the foundational work that enables this research.

References

S. Amland. Risk-based testing: Risk analysis fundamentals and metrics for software testing including a financial application case study. *Journal of Systems and Software*, 49(2-3):287–295, 1999.

Anthropic. The Claude 4.6 model family. Technical report, Anthropic, 2025.

D. Chicco and G. Jurman. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(6), 2020.

W.-L. Chiang, L. Zheng, Y. Sheng, A. N. Angelopoulos, T. Li, D. Li, H. Zhang, B. Zhu, M. Jordan, J. E. Gonzalez, and I. Stoica. Chatbot Arena: An open platform for evaluating LLMs by human preference. In *International Conference on Machine Learning*, 2024.

V. R. Basili, G. Caldiera, and H. D. Rombach. The Goal Question Metric approach. *Encyclopedia of Software Engineering*, pages 528–532, 1994.

M. Bhatt, S. Chennabasappa, C. Nikolaidis, S. Wan, I. Evtimov, D. Gabi, D. Song, F. Ahmad, C. Aber, A. Terzis, et al. Purple Llama CyberSecEval: A secure coding benchmark for language models. *arXiv preprint arXiv:2312.04724*, 2023.

M. Bhatt, S. Chennabasappa, Y. Li, C. Nikolaidis, D. Song, S. Wan, F. Ahmad, C. Aschermann, Y. Chen, D. Deleo, et al. CyberSecEval 2: A wide-ranging cybersecurity evaluation suite for large language models. *arXiv preprint arXiv:2404.13161*, 2024.

M. Damie, M. B. Ertan, D. Essoussi, A. Makhanu, G. Peter, and R. Wensveen. TOSSS: A CVE-based software security benchmark for large language models. *arXiv preprint arXiv:2603.10969*, 2026.

Y. Ding, Y. Fu, O. Ibrahim, C. Sitawarin, X. Chen, B. Alomair, D. Wagner, B. Ray, and Y. Chen. Vulnerability detection with code language models: How far are we? In *International Conference on Software Engineering (ICSE)*, 2025. arXiv:2403.18624.

R. Fang, R. Bindu, A. Gupta, Q. Zhan, and D. Kang. LLM agents can autonomously hack websites. *arXiv preprint arXiv:2402.06664*, 2024.

J. Feiglin and G. Dar. SastBench: A benchmark for testing agentic SAST triage. *arXiv preprint arXiv:2601.02941*, 2025.

Z. Gao, H. Wang, Y. Zhou, W. Zhu, and C. Zhang. How far have we gone in vulnerability detection using large language models. *arXiv preprint arXiv:2311.12420*, 2023.

T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé III, and K. Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.

Google. Gemini 3: A family of highly capable multimodal models. Technical report, Google DeepMind, 2025.

S. Halder, S. Saxena, K. K. Shrish, and T. M. SecLens: Evaluating LLM vulnerability detection with CVE-grounded tasks. Technical report, 2026.

C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making: Methods and Applications*. Springer-Verlag, Lecture Notes in Economics and Mathematical Systems, Vol. 186, 1981.

ISO/IEC. ISO/IEC 25010:2011 — Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. International Standard, 2011.

C. E. Jiménez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Z. Li, C. Wang, Z. Liu, H. Wang, S. Li, and C. Gao. SecEval: A comprehensive benchmark for evaluating cybersecurity knowledge of foundation models. *arXiv preprint arXiv:2312.10505*, 2023.

H. Lee, Z. Zhang, H. Lu, and L. Zhang. SEC-bench: Automated benchmarking of LLM agents on real-world software security tasks. *arXiv preprint arXiv:2506.11791*, 2025.

Z. Li, S. Dutta, and M. Naik. IRIS: LLM-assisted static analysis for detecting security vulnerabilities. In *International Conference on Learning Representations*, 2025. arXiv:2405.17238.

P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

S. Lu, B. Choi, and J. Chen. SecVulEval: Benchmarking LLMs for real-world C/C++ vulnerability detection. *arXiv preprint arXiv:2505.19828*, 2025.

X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024. arXiv:2308.03688.

Y. Liu, L. Gao, M. Yang, Y. Xie, P. Chen, X. Zhang, and W. Chen. VulDetectBench: Evaluating the deep capability of vulnerability detection with large language models. *arXiv preprint arXiv:2406.07595*, 2024.

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) – Protein Structure*, 405(2):442–451, 1975.

M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 220–229, 2019.

OWASP Foundation. OWASP Top 10:2021. https://owasp.org/Top10/, 2021.

OpenAI. GPT-5.4 system card. Technical report, OpenAI, 2025.

H. Pearce, B. Ahmad, B. Tan, B. Dolan-Gavitt, and R. Karri. Asleep at the keyboard? Assessing the security of GitHub Copilot's code contributions. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 754–768, 2022.

T. L. Saaty. How to make a decision: The analytic hierarchy process. *European Journal of Operational Research*, 48(1):9–26, 1990.

Z. Sheng, Z. Chen, S. Gu, H. Huang, G. Gu, and J. Huang. LLMs in software security: A survey of vulnerability detection techniques and insights. *arXiv preprint arXiv:2502.07049*, 2025.

M. L. Siddiq and J. C. S. Santos. SecurityEval dataset: Mining vulnerability examples to evaluate machine learning-based code generation techniques. In *Proceedings of the 1st International Workshop on Mining Software Repositories Applications for Privacy and Security*, pages 29–33, 2022.

W. Sun, J. Wang, Q. Guo, Z. Li, W. Wang, and R. Hai. CEBench: A benchmarking toolkit for the cost-effectiveness of LLM pipelines. *arXiv preprint arXiv:2407.12797*, 2024.

C. Tony, M. Mutas, N. T. Diem Ngo, and M. Ferrara. LLMSecEval: A dataset of natural language prompts for security evaluations. In *2023 IEEE/ACM International Conference on Mining Software Repositories (MSR)*, pages 588–592, 2023.

S. Ullah, M. Han, S. Pujar, H. Pearce, A. Coskun, and G. Stringhini. LLMs cannot reliably identify and reason about security vulnerabilities (yet?). In *IEEE Symposium on Security and Privacy (SP)*, 2024. arXiv:2312.12575.

M. Velasquez and P. T. Hester. An analysis of multi-criteria decision making methods. *International Journal of Operations Research*, 10(2):56–66, 2013.

S. Wan, C. Nikolaidis, D. Song, D. Molnar, J. Crnkovich, J. Grace, M. Bhatt, et al. CyberSecEval 3: Advancing the evaluation of cybersecurity risks and capabilities in large language models. *arXiv preprint arXiv:2408.01605*, 2024.

J. Zhang, C. Liu, K. Chen, and Z. Su. Repository-level vulnerability detection with LLMs: Challenges and opportunities. *arXiv preprint arXiv:2401.16185*, 2024.