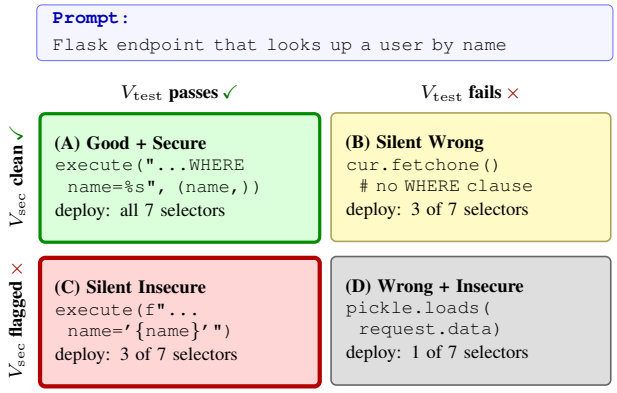
Selective Code Generation under Joint Correctness and Security Risk

Ahmad A. Rushdi [1]

Abstract

LLM-generated code can pass every unit test and still contain known-vulnerable patterns. Existing selective-generation methods — pass@1, best-of-K on tests, and one-objective conformal abstention — accept it silently. We extend Conformal Risk Control (CRC) to the joint event of accepting code that fails a test verifier or triggers a static analyzer. Our wrapper, *joint-CRC*, samples K candidates per problem, clusters by joint test-pass / static-analysis-clean behavior, and accepts when the largest good cluster clears a calibrated threshold; one CRC call bounds the joint silent-failure rate at a user-chosen α. On thirteen function-level CWE prompts (K=8, two generators), joint-CRC accepts correct-and-clean code in $82.6 \pm 1.4\%$ of cases with no observed joint silent failures, while pass@1 and pass@1+Semgrep gate leak joint-bad code at 42% and 26%. A security-only CRC baseline achieves zero security silent failures yet still leaks 34% joint-bad code, directly showing that controlling either axis alone does not control the joint event. Our experiments demonstrate calibrated control of the verifier-defined joint event; controlling hidden correctness failures beyond the supplied tests requires a held-out test split, left to future work.



(a) Four representative agent outputs and their outcome cells.

	A	B	C	D
pass@1	A	B	C	D
pass@1 + Semgrep gate	A	B	C	D
best-of-K tests*	A	B	C	D
best-of-K good*	A	B	C	D
correctness-CRC	A	B	C	D
security-CRC	A	B	C	D
joint-CRC (ours)	A	B	C	D

* oracle: peeks at V_{test} labels at decision time.

(b) Selector decisions per cell (colored = accept; outline = abstain).

Figure 1. Four outcomes by verifier verdict (V_{test} pass/fail, V_{sec} clean/flagged). Single-axis selectors silently deploy in (B), (C), or (D); joint-CRC abstains on *Silent Insecure* (C) by construction (no test-passing-and-clean cluster). *Silent Wrong* (B) requires failures beyond the supplied test verifier; in our experiments V_{test} scores both clustering and correctness (§7), so (B) is empirically absent on accepted samples by construction.

1. Introduction

LLM coding agents are increasingly deployed in production (Pearce et al., 2025; Kapoor et al., 2026), and two orthogonal failure modes determine their deployment cost (Figure 1a). *Silent Wrong* (B): a test-passing patch that breaks on edge cases the agent never generated. *Silent Insecure* (C): a test-passing patch that triggers a static-analysis rule for a weakness pattern (SQL injection, missing authentication, etc) detectable by off-the-shelf tools but never surfaced to the user (Liu et al., 2026). Throughout, "insecure" denotes the verdict of a fixed static analyzer — Semgrep (Semgrep, Inc., 2025) on the CWE-top-25 ruleset of the SecureForge benchmark (Liu et al., 2026); the wrapper's guarantee is conditional on this analyzer, not on ground-truth security.

Even when frontier models are explicitly prompted to "write secure production code" and told which CWE classes will be evaluated, a substantial fraction of their test-passing rollouts is still Semgrep-flagged (Liu et al., 2026): safety fine-tuning does not address the benign deployment case. This mirrors a deeper structural problem: Kalai et al. (2025) trace

[1]Stanford University, Stanford, CA, USA. Correspondence to: Ahmad A. Rushdi <rushdi@stanford.edu>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

LLM hallucinations to evaluations that *reward confident guessing over uncertainty acknowledgement*, so the agent's deployment-time incentive is to bluff rather than abstain.

We provide a calibrated alternative — an abstention rule with a finite-sample bound on the *joint* silent-failure rate — where abstention trades coverage for calibrated risk rather than silently shipping bad code. Operationally, the wrapper is the agent communicating uncertainty: α is the developer's trust budget, abstention routes the candidate for human review, and acceptance clears a known-quality bar.

The existing selective-generation toolkit gives one-objective guarantees. Conformal selective generation (Quach et al., 2024; Mohri & Hashimoto, 2024; Abbasi Yadkori et al., 2024) controls a single nonconformity score on a single loss; none of these methods control the deployment-relevant quantity, *accept and (wrong or insecure)*.

We close that gap by extending Conformal Risk Control (CRC) (Angelopoulos et al., 2024) with a joint silent-failure loss $L_{\text{joint}}(\lambda; x) = \mathbf{1}\{\text{accept}(x; \lambda) \wedge (\text{wrong}(x) \vee \text{insecure}(x))\}$, calibrated by a single threshold $\hat{\lambda}$ on a held-out split. Under exchangeability, $\mathbb{E}[L_{\text{joint}}(\hat{\lambda}; x_{\text{test}})] \leq \alpha$ — no Bonferroni correction, no per-axis calibration, no weight access.

Sampling K candidates per problem makes the joint event tractable. A reliable agent should concentrate on a *deployable* behavior: its K resamples should agree on which tests pass *and* produce no static-analysis findings. We use the joint-good concentration score $c_{\text{good}}(x) \in \{0, 1/K, \ldots, 1\}$ — the largest joint cluster restricted to clusters that are both test-passing and Semgrep-clean — and accept iff $c_{\text{good}}(x) \geq \hat{\lambda}$. Raw joint concentration $c_{\text{raw}} = \max_j |\mathcal{C}_j|/K$, the size of the largest behavioural cluster $\mathcal{C}_j$ over the K samples, is not the right deployment score: an agent that is correctness-stable but uniformly statically flagged collapses to a single high-c_{raw} cluster and would be accepted under a threshold rule on c_{raw}. Restricting to good clusters surfaces this *stable-correct-but-insecure* regime deterministically.

We contribute (i) joint-objective CRC for selective code generation, formalizing the joint silent-failure loss and the c_{good} score that bounds it with the same finite-sample guarantee as single-objective CRC (§3); (ii) the *stable-correct-but-insecure* regime as a measurable failure mode invisible to pass@k, best-of-K, and correctness-only CRC but controlled by the joint loss (§3.1); and (iii) empirical evaluation on two generators across a function-level Common Weakness Enumeration (CWE) microbenchmark, a SecureForge-style stress test, and a security-irrelevant control corpus, comparing joint-CRC to five non-oracle baselines and an oracle upper bound (§5.1, §5.2, Appendix A).

2. Related Work

Two technical lineages converge in our method, and two perspective threads motivate it. The first is conformal selective generation: Conformal Risk Control (Angelopoulos et al., 2024) extends split conformal prediction (Vovk et al., 2005; Lei et al., 2018; Angelopoulos & Bates, 2021) to finite-sample bounds on monotone losses, with prior application to factual language modeling (Quach et al., 2024; Mohri & Hashimoto, 2024) and hallucination abstention (Abbasi Yadkori et al., 2024) — each targeting a single loss, not the joint event. The second is K-sample behavioral clustering: AlphaCode (Li et al., 2022) clusters by test-pass behavior, self-consistency (Wang et al., 2023) majority-votes, semantic uncertainty (Kuhn et al., 2023) clusters by entailment, and repeated-sampling scaling (Brown et al., 2024) maps the inference-compute regime we operate in. We compose K-sample clustering with CRC for calibrated abstention on the joint correctness + security event.

On the security side, static-analysis evaluation of LLM code dates to Pearce et al. (2025); subsequent benchmarks (Bhatt et al., 2023; Vero et al., 2025) expand CWE coverage. Hardening divides into white-box fine-tuning (He & Vechev, 2023; He et al., 2024; Xu et al., 2025) and prompt-level interventions; the closest is Liu et al. (2026), which uses MCMC failure-distribution discovery plus GEPA prompt optimization (Agrawal et al., 2026) to harden frontier-model system prompts, reducing vulnerability rates by up to 48%. Deployed inference-time guardrails like LlamaFirewall / CodeShield (Chennabasappa et al., 2025) block insecure outputs at decision time without an abstention-rate guarantee. All these are generation- or filter-side and compose with joint-CRC: a hardened generator wrapped in our calibrated rule inherits both the lower base rate *and* the finite-sample joint bound.

The calibrated-abstention framing is itself recent: Kapoor et al. (2026) document that test-passing agent code is often rejected by maintainers as production-unready, with reliability lagging capability, and Kalai et al. (2025) attribute hallucination persistence to evaluations that penalize calibrated abstention rather than reward it. Joint-CRC instantiates this prescription for code generation: a finite-sample bound on the joint event, with abstention as a first-class outcome.

The closest formal precedent is finite-sample control of vector-valued risks: Risk-Controlling Prediction Sets (Bates et al., 2021), Learn-Then-Test (Angelopoulos et al., 2025), Pareto Testing (Laufer-Goldshtein et al., 2023), and Conformal Decision Theory (Lekeufack et al., 2024) bound a vector $(R_1, \ldots, R_M)$ via family-wise-error correction, giving $M\alpha$ on the union. We instead bound a single *disjunctive* event with one CRC call at α, tighter than Bonferroni at the cost of per-axis marginals.

3. Method

Let $\mathcal{A}$ be a code-generation agent. Given a problem x, $\mathcal{A}$ samples a candidate program $y \sim \pi_{\mathcal{A}}(\cdot \mid x)$ from its conditional distribution. We assume access to 2 deterministic verifiers operating on y: a sandbox V_{test} that executes y against a fixed test bank $\mathcal{T}$ and returns the per-test bit vector $V_{\text{test}}(y) \in \{0,1\}^{|\mathcal{T}|}$; and a static analyzer V_{sec} — Semgrep on the Community Python CWE-top-25 ruleset (Semgrep, Inc., 2025) — returning the set of triggered rule identifiers $V_{\text{sec}}(y)$. The framework permits a disjoint probe/gold partition of $\mathcal{T}$ (probe half for clustering, gold half held out for evaluation); the experiments in this paper use the full test suite for both, so that the demonstrated guarantee is over verifier outputs rather than over correctness behaviour beyond the supplied tests. We make that explicit in §7.

For a candidate y, the joint behavior signature is $\phi(y) = (\phi_{\text{test}}(y), \phi_{\text{sec}}(y))$, where $\phi_{\text{test}}(y) = V_{\text{test}}(y) \in \{0,1\}^{|\mathcal{T}|}$ is the per-test bit vector and $\phi_{\text{sec}}(y) = \text{hash}(V_{\text{sec}}(y))$ is a canonical fingerprint of the (sorted) Semgrep finding set. Two candidates are jointly equivalent iff their signatures match. Drawing K samples $y_1, \ldots, y_K \sim \pi_{\mathcal{A}}(\cdot \mid x)$ and partitioning by ϕ yields equivalence classes $\mathcal{C}_1, \ldots, \mathcal{C}_J$ (with $J \le K$). We call a cluster $\mathcal{C}_j$ *good* iff its signature is $(\mathbf{1}^{|\mathcal{T}|}, \emptyset)$: every member passes every test in $\mathcal{T}$ *and* triggers no Semgrep rule. We compare three scores:

$$c_{\text{raw}}(x) = \tfrac{1}{K} \max_j |\mathcal{C}_j|, \tag{1}$$

$$c_{\text{secClean}}(x) = \tfrac{1}{K} \big|\{k : V_{\text{sec}}(y_k) = \emptyset\}\big|, \tag{2}$$

$$c_{\text{good}}(x) = \tfrac{1}{K} \max_j |\mathcal{C}_j| \cdot \mathbf{1}\{\mathcal{C}_j \text{ is good}\}. \tag{3}$$

c_{raw} is the size of the largest cluster but cannot distinguish a stable Good (A) outcome from a stable Silent Insecure (C) outcome — both collapse to a single high- concentration cluster. c_{secClean} counts Semgrep-clean samples but is blind to the test verifier. Only c_{good} is non-zero when the agent concentrates on a behavior that is *both* test-passing and statically clean, so we use it as the deployment score throughout.

Let $\mathcal{G}(x) = \{j : \mathcal{C}_j \text{ is good}\}$ be the index set of good clusters, and let $y^\star_{\text{good}}(x)$ denote a representative of the dominant good cluster when one exists — any $y_k \in \arg\max_{j \in \mathcal{G}(x)} |\mathcal{C}_j|$, smallest index by convention. Define the threshold-indexed accept indicator

$$\text{accept}_\lambda(x) = \mathbf{1}\{\mathcal{G}(x) \neq \emptyset \wedge c_{\text{good}}(x) \ge \lambda\}, \tag{4}$$

so the wrapper abstains for every λ when $\mathcal{G}(x) = \emptyset$ (no test-passing-and-clean cluster exists), matching Algorithm 1. When $\mathcal{G}(x) \neq \emptyset$, define the indicators on the deployed representative $y^\star_{\text{good}}(x)$: $\text{wrong}(x) = \mathbf{1}\{y^\star_{\text{good}}(x) \text{ fails some } t \in \mathcal{T}\}$ (scored on the same $\mathcal{T}$ used for clustering, in the present experiments), $\text{insecure}(x) = \mathbf{1}\{V_{\text{sec}}(y^\star_{\text{good}}(x)) \neq \emptyset\}$, and $\text{bad}(x) = \text{wrong}(x) \vee \text{insecure}(x)$. The joint silent-failure loss is

$$L_{\text{joint}}(\lambda; x) = \mathbf{1}\{\text{accept}_\lambda(x) \wedge \text{bad}(x)\}, \tag{5}$$

with $L_{\text{joint}}(\lambda; x) = 0$ whenever $\mathcal{G}(x) = \emptyset$ (no accept, so no silent failure). L_{joint} is monotone in λ for fixed x. A single CRC call on a held-out calibration set $\mathcal{D}_{\text{cal}}$ of size n selects

$$\hat{\lambda} = \min\Big\{\lambda : \tfrac{1}{n} \sum_{x_i \in \mathcal{D}_{\text{cal}}} L_{\text{joint}}(\lambda; x_i) \le \alpha - \tfrac{1}{n}\Big\}, \tag{6}$$

and we accept x at test time iff $\text{accept}_{\hat{\lambda}}(x) = 1$. Under exchangeability of $\mathcal{D}_{\text{cal}}$ and test items, $\mathbb{E}[L_{\text{joint}}(\hat{\lambda}; x_{\text{test}})] = \Pr(\text{accept} \wedge \text{bad}) \le \alpha$ — we call the empirical version of this rate the *joint silent-failure rate* Sil_j throughout. The guarantee is on the joint event; neither marginal $\Pr(\text{accept} \wedge \text{wrong})$ nor $\Pr(\text{accept} \wedge \text{insecure})$ is separately controlled. Under the calibration-time analyzer, $\text{insecure}(x) = 0$ whenever $c_{\text{good}}(x) > 0$ (the deployed cluster is Semgrep-clean by construction), and in the present experiments $\text{wrong}(x) = 0$ on accepted samples also by construction since V_{test} scores both the clustering signature and correctness (§7). A held-out or secondary analyzer, or a held-out gold-test split, can break either constraint (Appendices B, C, D).

Three CRC modes share the wrapper above but calibrate different score/loss pairs. *Correctness-only CRC* accepts iff $c_{\text{raw}} \ge \hat{\lambda}_c$ against $L_c = \mathbf{1}\{\text{accept} \wedge \text{wrong}\}$; *security-only CRC* accepts iff $c_{\text{secClean}} \ge \hat{\lambda}_s$ against L_s (the analogous insecure-only loss); and *joint-CRC* (ours) accepts iff $c_{\text{good}} \ge \hat{\lambda}_j$ against L_{joint}. A Bonferroni composition of the two marginal CRCs gives 2α on the joint event; Joint-CRC controls the same event at α with one threshold and one calibration call (Figure 1b, Algorithm 1).

Algorithm 1 JOINT-CRC wrapper (test time)

Require: Agent $\mathcal{A}$, problem x, sample budget K, verifiers V_{test} and V_{sec}, calibrated threshold $\hat{\lambda}_j$

Sample $y_1, \ldots, y_K \sim \pi_{\mathcal{A}}(\cdot \mid x)$
Compute $\sigma_k = (V_{\text{test}}(y_k), \text{hash}(V_{\text{sec}}(y_k)))$ for $k = 1, \ldots, K$
Partition $\{y_k\}$ by σ_k into clusters $\mathcal{C}_1, \ldots, \mathcal{C}_J$
$\mathcal{G} \leftarrow \{j : \sigma_j = (\mathbf{1}^{|\mathcal{T}|}, \emptyset)\}$ {good-cluster indices}
if $\mathcal{G} = \emptyset$ **then**
  return ABSTAIN {$c_{\text{good}}(x) = 0$; no deployable cluster}
end if
$j^\star \leftarrow \arg\max_{j \in \mathcal{G}} |\mathcal{C}_j|$; $c_{\text{good}}(x) \leftarrow |\mathcal{C}_{j^\star}|/K$ {dominant good cluster, joint-good score}
if $c_{\text{good}}(x) \ge \hat{\lambda}_j$ **then**
  return representative $y_k \in \mathcal{C}_{j^\star}$ {deploy}
else
  return ABSTAIN {route x for human review}
end if

3.1. The stable-correct-but-insecure diagnostic

We define the *stable-correct-but-insecure* (SCI) indicator: $\mathrm{SCI}(x) = \mathbf{1}\{c_{\mathrm{raw}}(x) = 1 \wedge \neg\mathrm{wrong}(x) \wedge \mathrm{insecure}(x)\}$ — the K-sample-concentrated (c_{raw}=1) subset of the (C) Silent Insecure cell (Figure 1). Joint-CRC abstains on pure (C) clusters by construction ($c_{\mathrm{good}} = 0$ whenever every K-sample is Semgrep-flagged); on mixed cells with some clean and some flagged candidates, the wrapper deploys from the test-passing-and-clean cluster when $c_{\mathrm{good}} \geq \hat{\lambda}_j$. c_{raw}-thresholded and correctness-only methods are fooled specifically by SCI cases because they accept on $c_{\mathrm{raw}} = 1$. We report $\Pr(\mathrm{SCI} = 1)$ empirically in §5.1.

4. Experimental Setup

The primary evaluation corpus is a set of 13 function-level CWE microbenchmarks covering common Python-CWE patterns (SQL/command/eval/exec injection, unsafe deserialization, weak hashing, hardcoded secrets, JWT `none` algorithm, deprecated TLS; full list in Appendix E). Each prompt is paired with a pytest test bank that serves as V_{test} for both clustering and correctness scoring (see §7), and is constructed so that Semgrep's `auto` ruleset triggers on the canonical vulnerable pattern.

Two supporting corpora live in appendices: a smaller hand-curated mechanism-validation corpus (6 prompts $\rightarrow$ 24 cells; Appendix F) that confirms the same acceptance behavior in isolation, and a harder SecureForge-style web-app stress test (30 prompts on Flask/Jinja/PyMySQL harnesses) that we treat as a verifier-harness ceiling probe in §5.2. HumanEval+ (Liu et al., 2023) (Appendix G) is a 164-problem control where Semgrep findings are structurally rare, used to verify that Joint-CRC matches the best-of-K good oracle (which itself collapses to best-of-K tests) when $\mathrm{insecure}(x) \equiv 0$. Scaling calibration to the full SecureForge MCMC chain ($\sim$ 80K prompts) (Liu et al., 2026) and to SWE-chat OOD (Baumann et al., 2026) is left for follow-on work.

We deliberately evaluate two generators across a $\sim$100$\times$ parameter-scale gap: Claude Sonnet 4.6 (Anthropic, 2026) via API (frontier regime) and Qwen2.5-Coder-1.5B-Instruct (Hui et al., 2024) on M-series MPS in fp16 (small-local-reproducible regime).

Because joint-CRC's guarantee is on the calibrated decision rule rather than on the generator, the two cells serve as independent stress tests of the same theoretical claim — not a head-to-head comparison — and we report per-generator results in stacked sub-tables. Sampling uses K=8 candidates per problem, top-p=0.95, max-new-tokens=1024, seed 42; both generators run at $T \in \{0.4, 0.8\}$ on the primary corpus (yielding the $13 \times 2 \times 2$=52 cells reported in Table 1). The HumanEval+ control corpus (Appendix G) sweeps three temperatures on a single generator.

Functional correctness is scored by the LM-generated pytest unit tests released with each SecureForge scenario, holding the test bank constant per scenario and using the same V_{test} for both clustering and scoring (see §7). Security findings come from Semgrep 1.x Community Python ruleset (Semgrep, Inc., 2025) filtered to the CWE-top-25 subset used by Liu et al. (2026); each candidate runs in an isolated container with a 30 s wall-clock cap and disk-write disabled.

Calibration uses a 60/40 random calibration/test split with seed 42, 20 random splits per benchmark, mean and standard error reported. On the primary corpus (52 cells) this gives n_{cal}=31, n_{test}=21; on the stress corpus (120 cells) n_{cal}=72, n_{test}=48. All seven method comparisons are paired on identical samples within each temperature.

5. Experiments

We compare seven selective-generation methods on paired generated samples (Figure 2): two non-oracle baselines (pass@1 and pass@1 + Semgrep gate); two oracle upper bounds (best-of-K tests, which picks any test-passing sample, and best-of-K good, which picks any test-passing *and* Semgrep-clean sample — the deployment-equivalent ceiling on the joint event); two single-axis calibrated CRC variants (correctness-only on c_{raw}, security-only on c_{secClean}); and our joint-CRC, calibrated on c_{good}. The five K=8 methods (the two oracles and the three CRC variants) share the same candidate pool per cell, while pass@1 and pass@1+Semgrep gate consume only the first sample of that pool (K=1): the K-budget difference is part of the comparison, not noise we equalize. Oracle methods are reported only as upper bounds: they peek at V_{test} labels at decision time, which is exactly the quantity we want to bound silent failure against. All seven methods see identical test verifier outputs and identical generated samples per temperature, with generation amortized via batched decoding. The three experimental subsections that follow report (i) the headline result on the primary corpus (§5.1), (ii) a stress test that isolates verifier-harness ceiling effects (§5.2), and (iii) a per-generator breakdown that confirms the wrapper composes generator-agnostically (§5.3).

We report three outcome rates per method at $\alpha \in \{0.05, 0.10, 0.20\}$, partitioning the unit measure ($\mathrm{Abst} + \mathrm{Cov}_j + \mathrm{Sil}_j = 1$):

$$\mathrm{Abst} = \Pr(\neg\mathrm{acc}), \tag{7}$$

$$\mathrm{Cov}_j = \Pr(\mathrm{acc} \wedge \neg\mathrm{bad}), \tag{8}$$

$$\mathrm{Sil}_j = \Pr(\mathrm{acc} \wedge \mathrm{bad}). \tag{9}$$

Abst is the abstention rate; Cov_j is the joint coverage rate (fraction of items shipped that are *both* correct and Semgrep-clean); Sil_j is the joint silent-failure rate of Eq. 5, bounded at

α by construction for joint-CRC. Auxiliary metrics (single-axis silent failures, conditional answered rates, SCI rate, cost-family) are in Appendix H.

5.1. Main result

The primary corpus is 13 hand-curated function-level CWE microbenchmarks (Appendix E) chosen to satisfy three constraints: (i) every vulnerable reference reliably triggers Semgrep `auto`; (ii) every secure reference passes the test suite and is Semgrep-clean; (iii) tests exercise core behavior using stdlib or a single package, with no Flask/Jinja/PyMySQL lifecycle assumptions. Sweep design: 13 prompts $\times$ 2 generators $\times$ 2 temperatures $\times$ K=8 = 52 cells, n_{cal}=31 at the $60/40$ split (comfortably above the α=0.05 finite-sample floor). Table 1 reports the seven-method comparison at α=0.10 with mean $\pm$ SE across 20 random splits.

Joint-CRC achieves the headline at every tested $\alpha \in \{0.05, 0.10, 0.20\}$: $\text{Cov}_j = 82.6 \pm 1.4\%$, $\text{Sil}_j = 0.0 \pm 0.0\%$, $\text{Abst} = 17.4 \pm 1.4\%$ — column-for-column identical to the best-of-K good oracle. This identity is partly structural: V_{test} is the sole correctness verifier and defines both the clustering signature and $\text{wrong}(x)$, so accepted samples are test-passing by construction. Cell (B) Silent Wrong protection (failures *beyond* V_{test}) is therefore not demonstrated here; a held-out gold-test split would measure it (§7). The non-trivial empirical content is the cell (C) abstention: insecure events *can* occur on test-passing clusters, and joint-CRC sets $c_{\text{good}} = 0$ whenever every K-sample is Semgrep-flagged.

The non-oracle baselines, by contrast, ship verifier-defined joint-bad code at high rates: pass@1 at $\text{Sil}_j = 42.1 \pm 2.0\%$, pass@1 + Semgrep gate at $26.0 \pm 1.9\%$, and best-of-K on tests (a correctness oracle) at $15.0 \pm 1.2\%$, of which the stable-correct-but-insecure subset is the most concentrated diagnostic (§3.1). Each calibrated single-axis CRC variant satisfies its *own* bound but leaks heavily on the joint event, giving the cleanest possible "single-axis $\not\Rightarrow$ joint" demonstration: at α=0.10 correctness-CRC (calibrated on L_c) keeps $\text{Sil}_c = 4.5 \pm 2.0\% \leq \alpha$ but collapses to $\text{Abst} = 81.9\%$ and still leaves $\text{Sil}_j = 6.0 \pm 2.5\%$; security-CRC (calibrated on L_s) keeps $\text{Sil}_s = 0.0 \pm 0.0\% \leq \alpha$ at much lower abstention ($\text{Abst} = 2.4\%$, $\text{Cov}_j = 64.0\%$), but its Sil_j blows out to $33.6 \pm 2.0\%$ because controlling security marginal does not constrain the joint event. Joint-CRC is the *only score that separates verifier-defined deployable from non-deployable cells* under a single threshold.

The contrast is sharp: security-CRC hits Sil_s=0% within its bound but ships Sil_j=33.6%, while joint-CRC is the only deployable method that controls the joint verifier event at α while preserving useful Cov_j.

Table 1. Main result (primary corpus, α=0.10). Mean $\pm$ SE across 20 random $60/40$ calibration/test splits on 52 cells (13 function-level CWE prompts $\times$ 2 generators $\times$ 2 temperatures, K=8, n_{cal}=31). *Italic* rows are oracle upper bounds; **bold** is the deployable joint-CRC row. Each CRC variant is calibrated against its own loss (L_c, L_s, L_j respectively). Single-axis CRCs satisfy their own bound but leak on the joint event; joint-CRC matches the best-of-K good oracle on $(\text{Cov}_j, \text{Sil}_j)$.

Method (at α=0.10)	Abst	Cov_j	Sil_j
pass@1	0	57.9 ± 2.0	42.1 ± 2.0
pass@1 + Semgrep gate	16.2 ± 1.2	57.9 ± 2.0	26.0 ± 1.9
best-of-K tests (oracle)	8.6 ± 1.2	76.4 ± 1.5	15.0 ± 1.2
best-of-K good (oracle)	17.4 ± 1.4	$\mathbf{82.6 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$
correctness-CRC (L_c)	81.9 ± 7.4	12.1 ± 4.9	6.0 ± 2.5
security-CRC (L_s)	2.4 ± 0.7	64.0 ± 2.1	33.6 ± 2.0
joint-CRC (ours, L_j)	$\mathbf{17.4 \pm 1.4}$	$\mathbf{82.6 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$

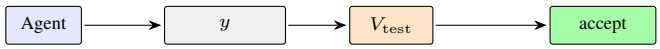
5.2. Stress test: verifier-harness ceiling

The primary corpus is controlled by design, which raises the obvious question of whether joint-CRC also holds up on harder material. We run an explicitly harder stress corpus to check this, and show that joint-CRC tracks the oracle upper bound rather than collapsing. The stress corpus is 30 SecureForge prompts generated by `sf generate --use-top-25` (covering 10 CWE classes, 120 cells total): Flask/PyMySQL/Jinja web-app scenarios with mock-heavy unittest fixtures.

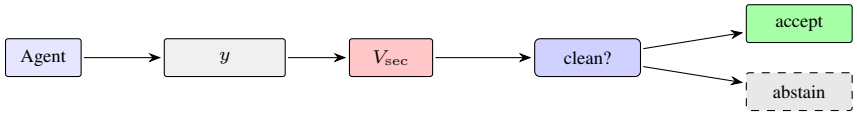
On this corpus the limiting factor is not the availability of Semgrep-clean code but compatibility with the generated test suites. Table 2 shows that at least one Semgrep-clean sample exists in 56.7% of cells (78.3% for Sonnet), whereas at least one test-passing sample exists in only 16.7%, and the intersection of both is just 2.5%. This is a correctness-verifier bottleneck driven by framework- and mock-binding brittleness, not a security-side ceiling; removing it in the primary corpus moves every existence probability up an order of magnitude (right column of Table 2). Even so, joint-CRC matches the best-of-K good oracle exactly: $\text{Cov}_j = 2.3\%$, $\text{Sil}_j = 0\%$, while pass@1 ships joint-bad code in 100% of accepted cells and pass@1+Semgrep-gate still leaves $\text{Sil}_j = 48.9\%$ (full table in Appendix I).

Table 2. Stress vs. primary corpus diagnostic. Per-cell "exists" rates and per-sample joint-good rate, pooled across generators and temperatures. Removing framework-level harness brittleness moves every existence probability up an order of magnitude — empirical basis for treating the SecureForge corpus as a verifier stress test (Appendix I) and the curated function-level corpus as the main suite for measuring useful coverage.

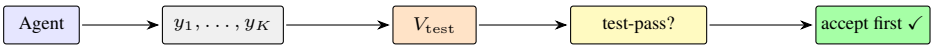
Quantity	stress	primary
$\Pr(\exists k$: test-pass)	16.7%	90.4%
$\Pr(\exists k$: Semgrep-clean)	56.7%	96.2%
$\Pr(\exists k$: good)	2.5%	82.7%
Per-sample good rate	0.6%	63.7%



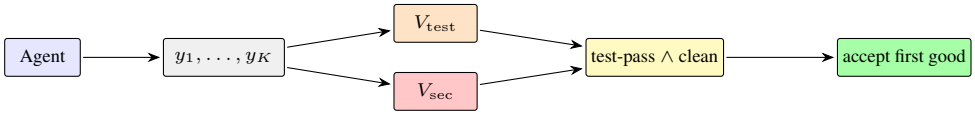
(a) [deployable] Pass@1 — deploy the single sample, no abstention.



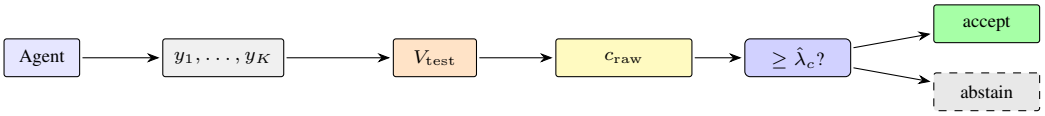
(b) [deployable] Pass@1 + Semgrep gate — single sample passes a deterministic security gate.



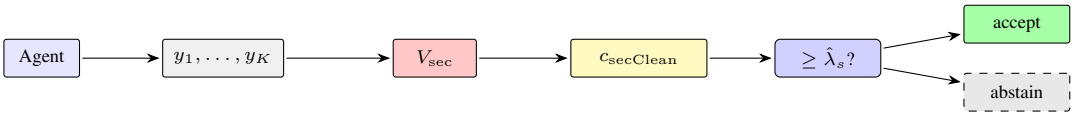
(c) [oracle, not deployable] Best-of-K tests — oracle picks any sample that passes V_{test}.



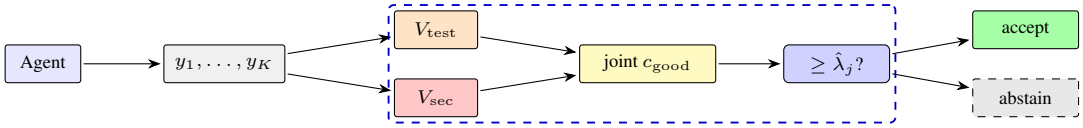
(d) [oracle, not deployable] Best-of-K good — oracle picks any sample that is test-passing *and* Semgrep-clean.



(e) [deployable] Correctness-CRC — calibrated threshold on c_{raw}; test-pass concentration.



(f) [deployable] Security-CRC — calibrated threshold on c_{secClean}; Semgrep-clean fraction.



(g) [deployable] **Joint-CRC (ours)** — calibrated threshold on the joint signature $(\phi_{\text{test}}, \phi_{\text{sec}})$.

Figure 2. Selector workflows: [deployable] decide using only signals available at inference time; [oracle, not deployable] peek at V_{test} labels at decision time and are reported only as upper bounds. Joint-CRC (g) is the only method that bounds $\Pr(\text{accept} \wedge (\text{wrong} \vee \text{insecure}))$.

The stress corpus thus functions as a verifier test rather than a method test: when the test suite rejects most plausible code for harness-binding reasons, even the best-of-K good oracle cannot deploy much, and joint-CRC tracks that ceiling rather than hallucinating safety. Both behaviors — low Cov_j when the verifier is brittle, and observed Sil_j within the α-bound (here empirically zero, stronger than required) — are exactly the behavior the bound is meant to induce. The primary corpus, designed to remove verifier brittleness, is therefore the appropriate setting for measuring useful coverage; the stress corpus measures the wrapper's behavior under verifier collapse.

5.3. Per-generator breakdown

The headline result pools across both generators; splitting it (Table 3) provides a "two witnesses" check across the $\sim$100$\times$ parameter-scale gap between Sonnet 4.6 and Qwen-1.5B (§4). The two generators differ dramatically in their pass@1 joint silent-failure rates — Sonnet $\text{Sil}_j = 23.0 \pm 2.2\%$, Qwen $\text{Sil}_j = 61.0 \pm 2.5\%$ — consistent with a roughly $2.6\times$ gap in joint reliability. After joint-CRC, *both* generators meet $\text{Sil}_j = 0.0 \pm 0.0\%$ and exactly match their respective best-of-K good oracle Cov_j: $94.5 \pm 1.4\%$ for Sonnet and $72.0 \pm 2.1\%$ for Qwen. The bound holds independently per generator, so joint-CRC composes black-box with whichever model is wrapped: the same calibration procedure adapts to each model's c_{good} distribution without retuning. Abstention is the price for the smaller, less reliable generator ($\text{Abst} = 28.0 \pm 2.1\%$ for Qwen vs $5.5 \pm 1.4\%$ for Sonnet), and that is the appropriate behavior: when a generator concentrates less mass on the good cluster, the calibrated threshold abstains more often rather than ship under the bound.

Three appendix studies reproduce the headline at smaller scale and stress the wrapper's analyzer-conditional limit. A mechanism-validation sweep on a 24-cell precursor (Ap-

Table 3. Per-generator breakdown (primary corpus, α=0.10). Pooled across $T \in \{0.4, 0.8\}$, 26 cells per generator, n_{cal}=15 at 60/40, 20 splits, mean ± SE. *Italic* rows are oracle upper bounds; **bold** is the deployable joint-CRC row. Each generator is an *independent* stress test of the bound, not a head-to-head comparison. Joint-CRC matches each generator's own best-of-K good oracle despite an ∼2.6× gap in their pass@1 joint silent-failure rates.

(a) Claude Sonnet 4.6.

Method	Abst	Cov_j	Sil_j
pass@1	0	77.0 ± 2.2	23.0 ± 2.2
pass@1 + Semgrep gate	7.0 ± 1.3	77.0 ± 2.2	16.0 ± 2.0
best-of-K tests (oracle)	5.5 ± 1.4	87.5 ± 1.8	7.0 ± 1.3
best-of-K good (oracle)	5.5 ± 1.4	94.5 ± 1.4	$\mathbf{0.0 \pm 0.0}$
correctness-CRC (L_c)	95.0 ± 5.0	3.5 ± 3.5	1.5 ± 1.5
security-CRC (L_s)	5.0 ± 1.4	79.0 ± 2.0	16.0 ± 2.0
joint-CRC (ours, L_j)	5.5 ± 1.4	$\mathbf{94.5 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$

(b) Qwen2.5-Coder-1.5B-Instruct.

Method	Abst	Cov_j	Sil_j
pass@1	0	39.0 ± 2.5	61.0 ± 2.5
pass@1 + Semgrep gate	21.0 ± 2.3	39.0 ± 2.5	40.0 ± 2.5
best-of-K tests (oracle)	14.0 ± 1.7	69.5 ± 2.1	16.5 ± 2.3
best-of-K good (oracle)	28.0 ± 2.1	72.0 ± 2.1	$\mathbf{0.0 \pm 0.0}$
correctness-CRC (L_c)	94.0 ± 4.4	3.0 ± 2.5	3.0 ± 2.1
security-CRC (L_s)	11.5 ± 2.2	41.5 ± 2.2	47.0 ± 2.3
joint-CRC (ours, L_j)	28.0 ± 2.1	$\mathbf{72.0 \pm 2.1}$	$\mathbf{0.0 \pm 0.0}$

pendix F) gives $\text{Cov}_j = 78.0 \pm 2.0\%$ and $\text{Sil}_j = 0.0 \pm 0.0\%$, matching the oracle ceiling on a corpus designed before final prompt selection and so free of selection effects. Phase E robustness perturbations (Appendix D) hold Sil_j at 0% under 10% Semgrep false positives, 10% gold-label flips, and line-number-sensitive cluster hashing; the only axis that degrades is 10% Semgrep false negatives, where Sil_j rises to 7.1% but tracks the oracle bound under the same corrupted signal — joint-CRC inherits the analyzer's blind spots rather than amplifying them. Finally, a HumanEval+ degeneration check (Appendix G) confirms that when $\text{insecure}(x) \equiv 0$, joint-CRC matches the best-of-K good oracle (which itself collapses to best-of-K tests), so the wrapper is strictly an additive guarantee over the correctness-only deployment baseline.

6. Computational complexity

Joint-CRC draws K samples instead of 1 and then computes a single decision from their joint signatures, so its overhead breaks down into generation cost, verifier cost, and a constant decision step. Generation dominates by an order of magnitude: at K=8, median end-to-end wall-clock per problem on the primary corpus is 25.9 s for Sonnet 4.6 (∼ 3.2 s per sample, dominated by API round-trip) and 15.0 s for Qwen-1.5B running fp16 on M-series MPS (∼ 1.9 s per sample). Static analysis is a fixed-overhead post-process per candidate (≤ 0.4 s/sample at Semgrep `auto`), and the cluster-and-threshold step is $O(K)$. In the deployment cost family of Appendix H, this gives a $/good (seconds per accepted-and-good output) that is ∼ 17× the $/correct of pass@1 but bounded; with α the only knob between guarantee and coverage, the operator picks the silent-risk budget directly instead of tuning a temperature or sampling ratio empirically.

Choosing K trades resolution of c_{good} against generation cost. Empirically, the marginal information saturates well below K=10: re-evaluating the same 52 cells at $K \in \{2, 4, 8\}$ gives joint-CRC $\text{Cov}_j = 72.9 \pm 1.5\%$ at K=2, $81.0 \pm 1.3\%$ at K=4, and $82.6 \pm 1.4\%$ at K=8, with Sil_j pinned to $0.0 \pm 0.0\%$ across all three budgets (Appendix J). Cov_j saturates between K=4 and K=8, consistent with the inference-compute-scaling regime mapped by Brown et al. (2024). Pass@1+Semgrep-gate is K-invariant by construction (it only ever sees the first sample), so its Sil_j stays at $26.0 \pm 1.9\%$ across the same sweep; the K=2→K=8 growth is exactly what unlocks joint-CRC's coverage advantage over gate-style baselines. We therefore use K=8 throughout as the practical operating point, noting that smaller-K deployments keep the bound at proportionally lower Cov_j.

7. Discussion

The methodological core of this paper is a single observation about CRC: it composes *disjunctively* for free. Any finite set of monotone silent-failure indicators $\{L_1, \ldots, L_M\}$ has a disjunctive composition $L_\vee = \mathbf{1}\{\bigvee_m L_m\}$ that is itself monotone in the threshold, so a single CRC call on $L_\vee$ bounds the union event at the same α (Angelopoulos et al., 2024), whereas a Bonferroni composition of M per-axis CRCs gives the weaker bound $M\alpha$. Joint-CRC is the special case M=2 with $L_1 = \text{accept} \wedge \text{wrong}$ and $L_2 = \text{accept} \wedge \text{insecure}$, but the construction extends directly to a typed-correctness verifier, a license checker, a performance-regression verifier, or any other deterministic monotone indicator. The reported $(\text{Abst}, \text{Cov}_j, \text{Sil}_j)$ triple treats abstention as neither success nor failure, aligning with Kalai et al. (2025)'s prescription that primary evaluations stop penalising uncertainty acknowledgement.

A second observation concerns where joint-CRC sits relative to generation-side hardening. SecureForge (Liu et al., 2026) reduces the base rate of vulnerable code by optimising the system prompt, while joint-CRC bounds the *conditional* accept-and-bad event regardless of the base rate, so the two interventions are orthogonal and compose: a hardened generator wrapped in joint-CRC inherits both the lower base rate *and* the calibrated abstention guarantee. On a deployment cost-vs-silent-failure Pareto plane, hardening shifts the curves downward and joint-CRC supplies a finite-sample-controlled operating point on each curve.

Four caveats bound the scope of the guarantee. First, and most important for honest scope: in the present experiments, V_{test} is the sole correctness verifier and is used both to define the clustering signature and to score wrong, so the bound is over the verifier-defined joint event, not over hidden correctness failures beyond the supplied test suite. Cell (B) Silent Wrong is empirically absent on accepted samples by this construction rather than by statistical bound; the framework permits a disjoint $\mathcal{T}_{\text{probe}} \sqcup \mathcal{T}_{\text{gold}}$ split that would lift this scope, and a held-out gold-test implementation is the natural next experiment. Second, the bound is *analyzer-conditional*: it holds with respect to Semgrep's labeling function, not against ground-truth security, so all claims here read as "statically-detectable." We quantify this dependence in three appendices — Phase E robustness under analyzer perturbations (Appendix D) shows graceful degradation under 10% Semgrep false negatives, the cross-analyzer audit (Appendix B) finds $9/13$ full agreement with an independent Bandit run, and the probe/gold split (Appendix C) calibrates on Semgrep but evaluates against Bandit as a held-out oracle and still retains $\text{Sil}_j = 0\%$. The wrapper is also single-language (Python; per-language Semgrep rulesets would extend it mechanically) and single-turn (multi-step agent trajectories like SWE-Agent or Aider would need per-decision CRC and are out of scope here), and depends on calibration-test exchangeability — the stress test of §5.2 exercises one shift (curated function-level → web-app SecureForge) as an exploratory probe rather than a finite-sample guarantee, and sharper distribution shifts would call for adaptive recalibration (Gibbs & Candès, 2021).

8. Conclusion

LLM coding agents fail silently in two orthogonal ways at deployment: they ship code that the supplied test verifier rejects, and they ship code that a static analyzer flags for a known weakness pattern. Existing selective-generation primitives control only one axis. Joint-CRC targets the deployment-relevant union event over those two verifiers with a single threshold and a single calibration call, exploiting CRC's disjunctive composability and a joint test-pass / static-analysis behavior signature. Its acceptance score c_{good} is non-zero only on test-passing, statically-clean clusters, so the wrapper abstains *by construction* on the stable-correct-but-insecure regime that is invisible to correctness-only evaluation. Empirical control of hidden correctness failures beyond the supplied test verifier — via a disjoint $\mathcal{T}_{\text{probe}} \sqcup \mathcal{T}_{\text{gold}}$ split — is the natural next experiment but is not part of the present submission (§7).

We frame the persuasive empirical result not as "Joint-CRC dominates" but as the following: it is the only *calibrated* method that controls the joint verifier event at α while preserving useful Cov_j, with a transparent cost / abstention tradeoff against cheap uncalibrated baselines (pass@1, pass@1 + Semgrep gate), the best-of-K good *oracle upper bound*, and single-axis CRC variants — each of which controls its own marginal loss but leaks heavily on the joint event (security-CRC at Sil_j=33.6% despite Sil_s=0%). In the human-centered framing of DL4C, the wrapper offers the developer a single steerable knob α for how much silent risk to tolerate, while the agent communicates uncertainty by abstaining — the alternative to bluffing that Kalai et al. (2025) call for in primary evaluations. Joint-CRC composes with generation-side hardening (Liu et al., 2026) and extends directly to more than two verifier objectives, so future work that adds a license checker, a typed-correctness verifier, or a performance-regression verifier slots into the same calibration scaffold. We release the wrapper, the synthetic and handwritten smoke tests, and the analysis scripts; the implementation is small enough — one Python package, a Semgrep config, a calibration script — for any team to add to an existing code-agent stack.

References

Abbasi Yadkori, Y., Kuzborskij, I., Stutz, D., György, A., Fisch, A., Doucet, A., Beloshapka, I., Weng, W.-H., Yang, Y.-Y., Szepesvári, C., Cemgil, A. T., and Tomasev, N. Mitigating LLM Hallucinations via Conformal Abstention, 2024.

Agrawal, L. A., Tan, S., Soylu, D., Ziems, N., Khare, R., Opsahl-Ong, K., Singhvi, A., Shandilya, H., Ryan, M. J., Jiang, M., Potts, C., Sen, K., Dimakis, A., Stoica, I., Klein, D., Zaharia, M., and Khattab, O. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2507.19457.

Angelopoulos, A. N. and Bates, S. A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification, 2021.

Angelopoulos, A. N., Bates, S., Fisch, A., Lei, L., and Schuster, T. Conformal Risk Control. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2208.02814.

Angelopoulos, A. N., Bates, S., Candès, E. J., Jordan, M. I., and Lei, L. Learn Then Test: Calibrating Predictive Algorithms to Achieve Risk Control. *Annals of Applied Statistics*, 19(2), 2025. doi: 10.1214/24-AOAS1998. arXiv:2110.01052.

Anthropic. Claude Sonnet 4.6 System Card. https://www.anthropic.com/claude-sonnet-4-6-system-card, 2026.

Bates, S., Angelopoulos, A. N., Lei, L., Malik, J., and Jordan, M. I. Distribution-Free, Risk-Controlling Prediction

Sets. *Journal of the ACM*, 68(6):43:1–43:34, 2021. doi: 10.1145/3478535. arXiv:2101.02703.

Baumann, J., Padmakumar, V., Li, X., Yang, J., Yang, D., and Koyejo, S. SWE-chat: Coding Agent Interactions From Real Users in the Wild, 2026.

Bhatt, M., Chennabasappa, S., Nikolaidis, C., Wan, S., Evtimov, I., Gabi, D., Song, D., Ahmad, F., Aschermann, C., Fontana, L., et al. Purple Llama CyberSecEval: A Secure Coding Benchmark for Language Models, 2023.

Brown, B., Juravsky, J., Ehrlich, R., Clark, R., Le, Q. V., Ré, C., and Mirhoseini, A. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling, 2024.

Chennabasappa, S., Nikolaidis, C., Song, D., Molnar, D., Ding, S., Wan, S., Whitman, S., Deason, L., Doucette, N., Montilla, A., Gampa, A., de Paola, B., Gabi, D., Crnkovich, J., Testud, J.-C., He, Y., Chaturvedi, R., Zhou, W., and Saxe, J. LlamaFirewall: An open source guardrail system for building secure AI agents, 2025.

Gibbs, I. and Candès, E. J. Adaptive Conformal Inference under Distribution Shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

He, J. and Vechev, M. Large Language Models for Code: Security Hardening and Adversarial Testing. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2023.

He, J., Vero, M., Krasnopolska, G., and Vechev, M. Instruction Tuning for Secure Code Generation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Hui, B., Yang, J., Cui, Z., Yang, A., Liu, D., Zhang, L., Zhang, T., Dang, K., Yu, B., Men, R., et al. Qwen2.5-Coder Technical Report, 2024.

Kalai, A. T., Nachum, O., Vempala, S. S., and Zhang, E. Why Language Models Hallucinate, 2025.

Kapoor, S., Kirgis, P., Schwartz, A., Rabanser, S., Allaire, J. J., Bommasani, R., Dubois, M., Hadfield, G., Hall, A., Hooker, S., Lazar, S., Newman, S., Papailiopoulos, D., Tekofsky, S., Toner, H., Ududec, C., and Narayanan, A. Open-World Evaluations for Measuring Frontier AI Capabilities. Preprint, https://cruxevals.com/open-world-evaluations.pdf, 2026.

Kuhn, L., Gal, Y., and Farquhar, S. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation. In *International Conference on Learning Representations (ICLR)*, 2023.

Laufer-Goldshtein, B., Fisch, A., Barzilay, R., and Jaakkola, T. Efficiently Controlling Multiple Risks with Pareto Testing. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.07913.

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., and Wasserman, L. Distribution-Free Predictive Inference for Regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.

Lekeufack, J., Angelopoulos, A. N., Bajcsy, A., Jordan, M. I., and Malik, J. Conformal Decision Theory: Safe Autonomous Decisions from Imperfect Predictions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. arXiv:2310.05921.

Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., Hubert, T., Choy, P., de Masson d'Autume, C., Babuschkin, I., Chen, X., Huang, P.-S., Welbl, J., Gowal, S., Cherepanov, A., Molloy, J., Mankowitz, D. J., Robson, E. S., Kohli, P., de Freitas, N., Kavukcuoglu, K., and Vinyals, O. Competition-Level Code Generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

Liu, H., Einstein, L., Yang, J., Baumann, J., Eddy, D., Manning, C. D., Kochenderfer, M., and Yang, D. SecureForge: Finding and Preventing Vulnerabilities in LLM-Generated Code via Prompt Optimization, 2026.

Liu, J., Xia, C. S., Wang, Y., and Zhang, L. Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Mohri, C. and Hashimoto, T. Language Models with Conformal Factuality Guarantees. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 36029–36047, 2024.

Pearce, H., Ahmad, B., Tan, B., Dolan-Gavitt, B., and Karri, R. Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions. *Communications of the ACM*, 68(2):96–105, 2025. doi: 10.1145/3610721.

Quach, V., Fisch, A., Schuster, T., Yala, A., Sohn, J. H., Jaakkola, T., and Barzilay, R. Conformal Language Modeling. In *International Conference on Learning Representations (ICLR)*, 2024.

Semgrep, Inc. Semgrep: Lightweight Static Analysis for Many Languages. Open-source static analysis tool, https://semgrep.dev/, 2025.

Vero, M., Mündler, N., Chibotaru, V., Raychev, V., Baader, M., Jovanović, N., He, J., and Vechev, M. BaxBench: Can LLMs Generate Correct and Secure Backends? In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic Learning in a Random World*. Springer, 2005.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations (ICLR)*, 2023.

Xu, X., Su, Z., Guo, J., Zhang, K., Wang, Z., and Zhang, X. ProSec: Fortifying code LLMs with proactive security alignment. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

A. Stable-correct-but-insecure rate per CWE

A stable-correct-but-insecure (SCI) cell satisfies $c_{\text{raw}}(x) = 1 \wedge \neg\text{wrong}(x) \wedge \text{insecure}(x)$: every $K{=}8$ sample passes the tests and triggers the same Semgrep finding. Any c_{raw}-thresholded or correctness-only baseline silently ships such code; joint-CRC abstains because $c_{\text{good}}{=}0$ on cell (C) of Figure 1. Table 4 counts SCI cells per CWE on the stress sweep; the primary corpus contains none.

Table 4. Per-CWE SCI rate on the stress sweep (120 cells, $K{=}8$, two generators, two temperatures). SCI concentrates on command-injection (CWE-78) and out-of-bounds-write (CWE-787); the primary corpus contains 0 observed SCI cells, since stable-flagged behavior is rare on stdlib-only single-file functions.

CWE	cells	SCI	rate
CWE-78 (cmd inj.)	16	2	12.5%
CWE-787 (OOB-W)	12	2	16.7%
others (CWE-22, -79, -89, -125, -352, -416, -434, -862)	92	0	0%
overall (stress)	120	4	3.3%
overall (primary)	52	0	0%

B. Cross-analyzer agreement

The bound is conditional on the supplied analyzer (§7). As a ground-truth audit, we run Bandit (MEDIUM/MEDIUM) on the 13 paired vulnerable/secure references (Appendix E) and compare its flagging decisions to Semgrep's (Table 5). Bandit reproduces Semgrep's "flag the vulnerable, clear the secure" decision on 9 of 13 tasks; the four disagreements come from Bandit's narrower rule coverage on hardcoded credentials, JWT `none` acceptance, formatted SQL, and deprecated TLS. Where both analyzers have a rule for the pattern they agree.

Table 5. Cross-analyzer agreement on the primary-corpus references. `Sg` = Semgrep `auto`; `Bd` = Bandit (MEDIUM/MEDIUM); counts are findings per analyzer. A task agrees when both flag the vulnerable reference and clear the secure one.

	vulnerable		secure		
Task	Sg	Bd	Sg	Bd	agree?
command_injection	1	1	0	0	✓
eval_injection	1	1	0	0	✓
exec_injection	1	1	0	0	✓
insecure_redirect	1	1	0	0	✓
marshal_unsafe	1	1	0	0	✓
tls_verify_off	1	1	0	0	✓
unsafe_deserialization	1	1	0	0	✓
weak_hash	1	1	0	0	✓
yaml_unsafe	1	1	0	0	✓
hardcoded_secret	1	0	0	0	–
jwt_none	1	0	0	0	–
sql_injection	2	0	0	0	–
weak_ssl	1	0	0	0	–
overall full agreement					$9/13 = 69\%$

C. Semgrep/Bandit security-oracle split

The headline result uses Semgrep as both the wrapper's probe (defining c_{good} and $\hat{\lambda}_j$) and the silent-failure oracle, so it is fair to ask whether $\text{Sil}_j{=}0\%$ is an artifact of that circularity. We re-ran the primary corpus with *both* Semgrep and Bandit findings recorded per sample (no new generation; one $K{=}8$ sweep into `medium_K8_dual.jsonl`). The wrapper still decides on Semgrep findings only, and we measure silent-failure under three independent oracles — Semgrep, Bandit, and "flagged by either" (Table 6). Joint-CRC keeps $\text{Sil}_j{=}0\%$ under all three.

This is non-trivial because the cross-analyzer audit (Appendix B) shows 31% disagreement on the "flag-the-vulnerable" label, yet the disagreement does not propagate to silent-failure on generated samples. Two reasons: joint-CRC only deploys from the dominant test-passing-and-Semgrep-clean cluster (which empirically also clears Bandit), and the disagreement is on the "flagged" side, not the "clean" side — analyzers agree on what is clean almost everywhere even when they disagree on which patterns to flag. The wrapper still inherits the analyzer-conditional caveat at the boundary (a sample insecure in the world but clean to both analyzers).

D. Robustness perturbations (Phase E)

We perturb cached smoke-sweep outputs along four independent axes and re-run the wrapper at a permissive deploy-anything-good threshold ($\hat{\lambda}_j{=}0^+$) on the perturbed labels, measuring silent-failure against the original (un-perturbed) labels. This isolates how label perturbations move the deployment decision, independent of calibration. Joint-CRC

Table 6. Semgrep/Bandit security-oracle split (primary corpus, α=0.10, 20 random 60/40 splits, mean ± SE). Wrapper decides on Semgrep findings; Sil_j scored against Semgrep, Bandit, and either-flag. Joint-CRC keeps Sil_j=0% under all three, matching the best-of-K good oracle column-for-column.

		Semgrep oracle		Bandit oracle (held-out)		either-flag oracle	
Method	Abst	Cov_j	Sil_j	Cov_j	Sil_j	Cov_j	Sil_j
pass@1	0	59.8 ± 2.1	40.2 ± 2.1	62.4 ± 2.0	37.6 ± 2.0	59.8 ± 2.1	40.2 ± 2.1
pass@1 + Semgrep gate	18.8 ± 1.3	59.8 ± 2.1	21.4 ± 1.7	59.8 ± 2.1	21.4 ± 1.7	59.8 ± 2.1	21.4 ± 1.7
best-of-K tests (oracle)	8.6 ± 1.2	73.8 ± 1.6	17.6 ± 1.2	79.3 ± 1.4	12.1 ± 1.1	73.8 ± 1.6	17.6 ± 1.2
best-of-K good (oracle)	17.4 ± 1.4	82.6 ± 1.4	0.0 ± 0.0	82.6 ± 1.4	0.0 ± 0.0	82.6 ± 1.4	0.0 ± 0.0
correctness-CRC (L_c)	81.2 ± 7.6	11.9 ± 4.8	6.9 ± 2.8	12.6 ± 5.0	6.2 ± 2.6	11.9 ± 4.8	6.9 ± 2.8
security-CRC (L_s)	2.9 ± 1.1	68.6 ± 2.0	28.6 ± 1.8	68.6 ± 2.0	28.6 ± 1.8	68.6 ± 2.0	28.6 ± 1.8
joint-CRC (ours, L_j)	$\mathbf{17.4 \pm 1.4}$	$\mathbf{82.6 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{82.6 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{82.6 \pm 1.4}$	$\mathbf{0.0 \pm 0.0}$

is invariant to 10% Semgrep false positives, 10% gold-label flips, and line-number-sensitive cluster hashing (Table 7). Only 10% Semgrep false negatives degrade it — Sil_j rises from 0% to 7.1%, but the oracle bound rises to 7.1% under the same injection: if the verifier suppresses a true finding, neither joint-CRC nor the oracle can identify the output as insecure.

Table 7. Phase E robustness: joint-CRC under 10% verifier-signal perturbations on the Layer-3 smoke set (42 cells, K=4). Only Semgrep false-negative injection raises Sil_j, and joint-CRC matches the (also-corrupted) oracle bound.

Perturbation	Cov_j	Sil_j	Abst
none (baseline)	69.0%	0.0%	31.0%
Semgrep 10% FP injection	69.0%	0.0%	31.0%
Semgrep 10% FN injection	69.0%	7.1%	23.8%
Gold 10% label flips	69.0%	0.0%	31.0%
Hash includes line numbers	69.0%	0.0%	31.0%

E. Primary corpus: prompt set

The 13 CWE microbenchmarks of the primary corpus. Each task is a single-file Python function with a small pytest bank using stdlib or one package, paired with a `vulnerable.py` reference (triggers Semgrep `auto`) and a `secure.py` reference (passes the tests, Semgrep-clean).

F. Mechanism-validation precursor sweep

A 24-cell precursor sweep run before the primary corpus: six handwritten function-level CWE microbenchmarks × two generators (Sonnet 4.6, Qwen2.5-Coder-1.5B) × two temperatures at K=8, calibrated on 20 random 60/40 splits. At $\alpha \in \{0.10, 0.20\}$, joint-CRC keeps Sil_j=0% at Cov_j=78.0 ± 2.0%, matching the best-of-K good oracle (Table 9); pass@1, pass@1+Semgrep gate, and best-of-K tests leak 33.5%, 25.5%, and 8.0% joint-bad — test-passing selection alone is not sufficient. At α=0.05 all three CRC methods abstain everything because n_{cal}=14 makes the finite-sample correction $\alpha - 1/n_{\mathrm{cal}}$ negative; the primary corpus (n_{cal}=31) recovers α=0.05 coverage.

Table 8. Primary-corpus prompts. Source: `code/dl4c/jointcrc/smoke_bench/`. Each row pairs a *vulnerable* reference (the natural mistake) with a *secure* reference (production-grade alternative).

CWE	Task	Vulnerable / secure
CWE-78	cmd injection	`shell=True` / argv list
CWE-89	SQL injection	f-string / parameterised
CWE-94	eval injection	`eval()` / `ast.literal_eval`
CWE-95	exec injection	`exec()` / AST walk
CWE-22	tar slip	`extractall` / member-path check
CWE-295	TLS verify off	`verify=False` / default verify
CWE-327	weak hash	MD5 / SHA-256
CWE-327	weak TLS	`TLSv1_1` / default context
CWE-345	JWT none	accepts `none` / `['HS256']`
CWE-502	unsafe pickle	`pickle.loads` / JSON
CWE-502	unsafe YAML	`yaml.load` / `safe_load`
CWE-502	unsafe marshal	`marshal.loads` / JSON
CWE-798	hardcoded secret	literal key / `os.environ`

Table 9. Mechanism-validation precursor sweep (α=0.10, 24 cells, K=8, 20 random 60/40 splits, mean ± SE). *Italic* rows are oracle upper bounds; **bold** is the deployable joint-CRC row.

Method	Abst	Cov_j	Sil_j
pass@1	0	66.5 ± 2.1	33.5 ± 2.1
pass@1 + Semgrep gate	8.0 ± 1.2	66.5 ± 2.1	25.5 ± 2.3
best-of-K tests (oracle)	14.0 ± 2.1	78.0 ± 2.0	8.0 ± 1.2
best-of-K good (oracle)	22.0 ± 2.0	78.0 ± 2.0	$\mathbf{0.0 \pm 0.0}$
correctness-CRC (L_c)	86.5 ± 7.4	8.0 ± 4.4	5.5 ± 3.1
security-CRC (L_s)	6.0 ± 1.7	65.0 ± 2.5	29.0 ± 2.2
joint-CRC (ours, L_j)	22.0 ± 2.0	$\mathbf{78.0 \pm 2.0}$	$\mathbf{0.0 \pm 0.0}$

G. HumanEval+ degeneration check

We re-run the wrapper on HumanEval+ (Liu et al., 2023) (164 algorithmic problems, no user input, I/O, DB, or network surface) using cached samples from Qwen2.5-Coder-1.5B-Instruct at K=8 across three temperatures. A Semgrep `auto` spot-check on 30 random (problem, sample) pairs at T=0.8 produced 0 findings, so $\mathrm{insecure}(x) \equiv 0$

on this corpus. Under that condition L_{joint} reduces to $\mathbf{1}\{\text{accept} \wedge \text{wrong}\}$ and c_{good} reduces to $1/K$ times the largest test-passing cluster — the best-of-K tests rule. Table 10 confirms this degeneration: joint-CRC matches the best-of-K good oracle at every temperature, so without a security signal joint-CRC delivers strictly the correctness-only guarantee.

Table 10. HumanEval+ degeneration check (α=0.10, 164 problems, K=8, 20 random 60/40 splits, mean ± SE). When $\text{insecure}(x) \equiv 0$, joint-CRC and best-of-$K$ good produce identical $(\text{Abst}, \text{Cov}_j, \text{Sil}_j)$ at every T; one column reported.

T	Abst	Cov_j	Sil_j
0.4	19.4 ± 0.5	80.6 ± 0.5	0.0 ± 0.0
0.8	16.2 ± 0.5	83.8 ± 0.5	0.0 ± 0.0
1.2	19.0 ± 0.7	81.0 ± 0.7	0.0 ± 0.0

H. Auxiliary metric definitions

Beyond the partition triple $(\text{Abst}, \text{Cov}_j, \text{Sil}_j)$ (Eqs. 7–9), appendix tables tabulate the following at $\alpha \in \{0.05, 0.10, 0.20\}$. Single-axis silent-failure rates $\text{Sil}_c = \Pr(\text{acc} \wedge \text{wrong})$, $\text{Sil}_s = \Pr(\text{acc} \wedge \text{insecure})$ are the empirical L_c, L_s from §3; bounded only by the matching single-axis CRC. Effective rates $\text{Eff} = \Pr(\text{acc} \wedge \neg\text{wrong})$ and $\text{Sec} = \Pr(\text{acc} \wedge \neg\text{insecure})$ are answered-and-correct / answered-and-clean fractions; C-Acc, C-Sec are their answered-only conditionals. $\text{SCI} = \Pr(c_{\text{raw}}{=}1 \wedge \neg\text{wrong} \wedge \text{insecure})$ is the stable-correct-but-insecure rate (§3.1). The cost family: Wall is end-to-end median wall-clock per problem; Gen and Verify isolate generation and verifier cost; $\$/\text{correct} = \text{Wall}/\text{Eff}$, $\$/\text{good} = \text{Wall}/\text{Cov}_j$, and K-overhead $= \text{Wall}/\text{Wall}_{K=1}$.

I. Stress corpus: per-CWE detail

On the 120-cell SecureForge stress corpus (§5.2; Flask/PyMySQL/Jinja web apps with mock-heavy unittest fixtures), attainable joint coverage is very low: even best-of-K good reaches Cov_j=2.3% (Sil_j=0%), and joint-CRC matches that ceiling while pass@1 ships joint-bad code in 100% of accepted cells and pass@1+Semgrep-gate leaves Sil_j=48.9%. The 2.3% ceiling is test-brittleness driven, not a security ceiling: a Semgrep-clean sample exists in most cells, but a test-passing sample exists in under 20%, and the intersection is essentially zero outside memory-safety CWEs — a property of the harness, not the wrapper.

Concretely (`any-g` = some K-sample is test-passing *and* clean): CWE-787 (OOB-W, 12 cells) yields `any-g`=8.3%, CWE-125 (OOB-R, 12 cells) yields `any-g`=16.7%; the other eight CWEs yield `any-g`=0%, and the overall stress-corpus rate is 2.5% across 120 cells.

J. K-sensitivity ablation

Subsampling each primary-corpus cell to $K \in \{2, 4, 8\}$ (first K samples by stored order; 20 random 60/40 splits per K, α=0.10, mean ± SE). Joint-CRC and best-of-K good agree column-for-column at every K. Cov_j rises from 72.9±1.5% at K=2 to 81.0±1.3% at K=4 to 82.6±1.4% at K=8 and saturates visibly between K=4 and K=8, consistent with Brown et al. (2024)'s $K \approx 10$ point; Sil_j holds at 0% across the sweep. Pass@1+Semgrep-gate stays K-invariant at Sil_j=26% (it sees only the first sample), so the K=2→K=8 growth in joint-CRC's Cov_j is exactly what unlocks its coverage advantage over gate-style baselines. Practically, K=8 is the smallest budget that captures the saturated regime without paying for the marginal improvement at $K \geq 10$.