

simplecov

github.com/simplecov-ruby/simplecov

Scanned 24 September 2026 at 09:43 · Report generated 29 October 2026 · Ruby · 4,926 stars

OVERALL ASSESSMENT

High

Combined score 121 · 47 findings after filtering

Summary

This repository has a high vulnerability score of 119.0, which means there are several common coding mistakes that could be exploited by attackers if the code is used in production. While the malicious-pattern score is very low at 2.0, suggesting the code itself is not intentionally harmful, the high vulnerability score indicates that many issues stem from standard insecure practices rather than hidden malware. The most critical finding is a controller missing cross-site request forgery protection, which could allow an attacker to trick users into visiting malicious sites. Additionally, several commands like `system`, `spawn`, and `Open3` are used without verifying user input, creating potential for code injection attacks if bad data reaches these functions. There are also some concerning issues with HTML escaping and sensitive information exposure that should be addressed before the project is deployed to a live environment.

Written by a language model running locally on the scanning machine, from the findings in this report. The risk levels above and below were decided by the scoring model, not by this summary.

Risk, by type

MalDet scores two different questions separately, because they call for different responses: code can be badly written without being hostile, and hostile code is not always badly written.

VULNERABILITY

High (119)

Insecure coding an attacker could exploit. Usually mistakes, and usually fixable in place.

MALICIOUS PATTERN

Low (2)

Signs the code may be trying to do harm: hidden payloads, credential access, persistence, exfiltration.

What each detector found

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
semgrep	Insecure coding patterns across many languages	5	40	0	45
guarddog	Malicious package and install-time behaviour	0	0	2	2

47 of 47 findings were rated more likely genuine than noise by the classifier.

Evidence

Identical findings are grouped, so a single rule firing many times takes one row. Ordered worst first. “Seen” is how many times that finding occurred.

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Detected controller which does not enable cross-site request forgery protections using 'protect_from_forgery'. Add 'protect_from_forgery :with => :exception' to your controller class.	1	/test_projects/rails/rspec_rails/app/controllers/application_controller.rb:1
HIGH	semgrep	This GitHub Actions workflow file uses `pull_request_target` and checks out code from the incoming pull request. When using `pull_request_target`, the Action runs in the context of the target repository, which includes access to all repository secrets. Normally, this is safe because the Action only	1	/.github/workflows/dependabot_as_sets.yml:38
HIGH	semgrep	Using variable interpolation `\${{...}}` with `github` context data in a `run:` step could allow an attacker to inject their own code into the runner. This would allow them to steal secrets and code. `github` context data can have arbitrary user input and should be treated as untrusted. Instead, use	3	/.github/workflows/mutation.yml:65
MEDIUM	semgrep	Detected non-static command inside system. Audit the input to 'system'. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	3	/lib/simplecov/cli/open.rb:20

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected non-static command inside Open3.capture3. Audit the input to 'Open3.capture3'. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	2	/spec/production_integration_spec.rb:18
MEDIUM	semgrep	Detected non-static command inside spawn. Audit the input to 'spawn'. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	2	/lib/simplecov/cli/watch.rb:55
MEDIUM	semgrep	Detected non-static command inside Open3.popen2e. Audit the input to 'Open3.popen2e'. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	1	/spec/support/sandbox_project.rb:65
MEDIUM	semgrep	Should not use SHA1 to generate hashes. There is a proven SHA1 hash collision by Google, which could lead to vulnerabilities. Use SHA256, SHA3 or other hashing functions instead.	2	/spec/sandbox/file_id_collisions_spec.rb:46
MEDIUM	semgrep	Detected an element with disabled HTML escaping. If external data can reach this, this is a cross-site scripting (XSS) vulnerability. Ensure no external data can reach here, or remove 'escape=false' from this element.	1	/lib/simplecov/formatter/html_formatter/public/index.html:69
MEDIUM	semgrep	Found that the setting for providing detailed exception reports in Rails is set to true. This can lead to information exposure, where sensitive system or internal information is displayed to the end user. Instead, turn this setting off.	4	/test_projects/rails/rspec_rails/config/environments/development.rb:15

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected non-static command inside `...`. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	2	/benchmarks/report_scale.rb:61
MEDIUM	semgrep	GitHub Actions step uses a mutable tag or branch reference. Tags and branch names can be silently repointed by the action owner, enabling supply-chain attacks — as seen in the trivy-action and kics-github-action compromises. Pin the reference to a full 40-character commit SHA instead, e.g. `uses: ac	23	/.github/workflows/unstable.yml:35
LOW	guarddog	Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]	2	/spec/simple_cov/cli_spec.rb:8985

How to read this report

- **This is a static scan.** The code was read, never executed. Behaviour that only appears at runtime is outside what any of these detectors can see.
- **Findings are not all equally real.** A classifier rates each finding on how likely it is to be genuine rather than noise, and that rating weights the scores. Measured against a hand-reviewed sample it reached precision 0.52 and recall 0.62 — useful for ranking, not a verdict on any single finding.
- **A high score is not proof of malice.** Security tools, penetration-testing frameworks and malware research repositories contain the same patterns as real malware by design. The two risk types above exist to keep that distinction visible.
- **Absence of findings is not proof of safety.** A detector that is not installed, or a language none of them parse, produces silence rather than a warning.
- **This scan is a snapshot.** The repository may have changed since 24 September 2026.