

mitamae

github.com/itamae-kitchen/mitamae

Scanned 24 September 2026 at 10:25 · Report generated 29 October 2026 · Ruby · 390 stars

OVERALL ASSESSMENT

Medium

Combined score 33 · 19 findings after filtering

Summary

This repository has a medium vulnerability score of 33, which means there are some insecure coding patterns that an attacker could exploit if they gain access to the code. However, the scan found no signs of malicious intent or hidden malware, so the code is not actively dangerous on its own. The main issues are standard mistakes like using non-static commands in IO.popen and system, which can be exploited if user data isn't properly validated before running them. There are also some configuration gaps, such as missing a cooldown period for Dependabot updates and the use of mutable GitHub Actions tags that could theoretically lead to supply-chain attacks. While these findings look like ordinary insecure coding rather than malicious intent, they do not make the code safe to use without further review and fixes.

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

Medium (33)

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

MALICIOUS PATTERN

Safe (0)

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	0	18	0	18

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
yara	Signatures of malicious behaviour in source files	0	0	1	1

18 of 19 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
MEDIUM	semgrep	Detected non-static command inside IO.popen. Audit the input to 'IO.popen'. 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/spec_helper.rb:47
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.	1	/build_config.rb:20
MEDIUM	semgrep	This Dependabot configuration does not set a cooldown period. Newly published packages can be malicious or unstable. Add a `cooldown` block with `default-days: 7` to each `package-ecosystem` entry under `updates` to wait 7 days before proposing updates to newly published package versions. Reference:	1	/.github/dependabot.yml:3
MEDIUM	semgrep	Detected manual creation of an ERB template. Manual creation of templates may expose your application to server-side template injection (SSTI) or cross-site scripting (XSS) attacks if user input is used to create the template. Instead, create a '.erb' template file and use 'render'.	1	/mrblib/mitamae/resource_executor/template.rb:52

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
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	14	/.github/workflows/build.yml:21
LOW	yara	YARA rule matched: detect_downloader — Downloads a remote resource - common and low-signal alone; see detect_download_and_execute for the composite version	1	/README.md

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.