

FreshRSS

github.com/FreshRSS/FreshRSS

Scanned 24 September 2026 at 07:56 · Report generated 29 October 2026 · PHP · 16,126 stars

OVERALL ASSESSMENT

High

Combined score 103 · 108 findings after filtering

Summary

This repository has a high vulnerability score of 103.0, which means there are many insecure coding patterns that an attacker could exploit to break into the system or steal data. While the malicious-pattern score is very low at 1.0, suggesting the code does not appear to be intentionally harmful, the high vulnerability score indicates that the code contains common mistakes like using weak passwords and executing commands with user input. The scan also found several medium-risk issues, such as potential cross-site scripting vulnerabilities and overly permissive security headers, which could allow attackers to bypass protections or inject malicious scripts into the application. Although some findings are flagged as potentially malicious by automated tools, most of them stem from standard insecure practices rather than deliberate attacks, so they should be addressed through careful code review and updates to fix the underlying weaknesses before deploying the software.

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 (103)

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

MALICIOUS PATTERN

Low (1)

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	4	78	0	82
yara	Signatures of malicious behaviour in source files	14	0	2	16
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	10	10

28 of 108 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	bcrypt hash detected	1	/app/Utils/passwordUtil.php:33
HIGH	semgrep	Executing non-constant commands. This can lead to command injection.	2	/lib/phpmailer/phpmailer/src/PHPMailer.php:1969
HIGH	semgrep	Evaluating non-constant commands. This can lead to command injection.	1	/cli/i18n/I18nFile.php:85
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	4	/.github/workflows/tests.yml
HIGH	yara	YARA rule matched: detect_obfuscated_string_execution — Reconstructs a string character-by-character or from hex escapes, then immediately executes it - fromCharCode/chr() alone are common in ordinary text-processing code, so this requires both the reconstruction and an execution call together	2	/app/Utils/httpUtil.php

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	8	/Docker/entrypoint.sh
MEDIUM	semgrep	File name based on user input risks server-side request forgery.	14	/p/f.php:69
MEDIUM	semgrep	Make sure comparisons involving md5 values are strict (use `===` not `==`) to avoid type juggling issues	12	/app/Utils/dotNotationUtil.php:225
MEDIUM	semgrep	Access-Control-Allow-Origin response header is set to "*". This will disable CORS Same Origin Policy restrictions.	2	/p/api/greader.php:1091
MEDIUM	semgrep	Detection of insertAdjacentHTML from non-constant definition. This can inadvertently expose users to cross-site scripting (XSS) attacks if this comes from user-provided input. If you have to use insertAdjacentHTML, consider using a sanitization library such as DOMPurify to sanitize your HTML.	7	/p/scripts/main.js:1834
MEDIUM	semgrep	Calling `unserialize()` with user input in the pattern can lead to arbitrary code execution. Consider using JSON or structured data approaches (e.g. Google Protocol Buffers).	6	/lib/simplepie/simplepie/src/Cache/File.php:88
MEDIUM	semgrep	Using user input when deleting files with `unlink()` is potentially dangerous. A malicious actor could use this to modify or access files they have no right to.	37	/app/Models/FormAuth.php:92
LOW	dep_checker	Loose version constraint: 'stylelint-order^8.1.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@eslint/js^10.0.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'stylelint^17.15.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'stylelint-config-recommended^18.0.0' — attacker could push malicious update	1	package.json

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	dep_checker	Loose version constraint: 'rtlcss^4.3.0' — attacker could push malicious update	1	package.js on
LOW	dep_checker	Loose version constraint: 'neostandard^0.14.0-next.1' — attacker could push malicious update	1	package.js on
LOW	dep_checker	Loose version constraint: 'markdownlint-cli2^0.23.2' — attacker could push malicious update	1	package.js on
LOW	dep_checker	Loose version constraint: 'globals^17.12.0' — attacker could push malicious update	1	package.js on
LOW	dep_checker	Loose version constraint: 'eslint^10.10.0' — attacker could push malicious update	1	package.js on
LOW	dep_checker	Loose version constraint: '@stylistic/stylelint-plugin^5.3.0' — attacker could push malicious update	1	package.js on
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	2	/.github/ workflows/ tests.yml

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.