

livewire

github.com/livewire/livewire

Scanned 24 September 2026 at 09:14 · Report generated 29 October 2026 · PHP · 23,582 stars

OVERALL ASSESSMENT

High

Combined score 96 · 81 findings after filtering

Summary

This repository has a high vulnerability score of 96.0, which means an attacker could easily exploit insecure coding patterns to break your application or steal data. Most of the findings are standard security issues like missing CSRF protection and potential path traversal bugs that can be fixed by updating code or adding checks. The scan did not find any signs of malicious intent, so these problems appear to be ordinary mistakes rather than deliberate attacks. However, there are several low-priority issues related to outdated package versions that could introduce new risks if the library owners release dangerous updates in the future. You should address the critical and medium-severity findings first by reviewing and fixing the code examples provided in the report.

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 (96)

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	61	4	65

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	16	16

43 of 81 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	RegExp() called with a `attr` function argument, this might allow an attacker to cause a Regular Expression Denial-of-Service (ReDoS) within your application as RegExp blocks the main thread. For this reason, it is recommended to use hardcoded regexes instead. If your regex is run on user-controlled	1	/js/plugins/navigate/page.js:225
MEDIUM	semgrep	The application appears to have disabled the CSRF token verification middleware which can lead to a Cross-site request forgery (CSRF) vulnerability. If the application uses cookie-based authentication, an attacker can trick users into sending authenticated HTTP requests without their knowledge from	1	/legacy_tests/HttpKernel.php:29
MEDIUM	semgrep	The application appears to have disabled the cookie encryption middleware which can lead to a Cross-site request forgery (CSRF) vulnerability. If the application uses cookie-based authentication, an attacker can trick users into sending authenticated HTTP requests without their knowledge from any a	1	/legacy_tests/HttpKernel.php:29
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	40	/.github/workflows/test-prs.yml:242

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	The <code>`redirect()`</code> method does not check its destination in any way. If you redirect to a URL provided by end-users, your application may be open to the unvalidated redirects security vulnerability. Consider using literal values or an allowlist to validate URLs.	14	/src/Features/SupportRedirects/HandlesRedirects.php:34
MEDIUM	semgrep	Detected possible user input going into a <code>`path.join`</code> or <code>`path.resolve`</code> function. This could possibly lead to a path traversal vulnerability, where the attacker can access arbitrary files stored in the file system. Instead, be sure to sanitize or validate user input first.	1	/scripts/build.js:109
MEDIUM	semgrep	Using user input when deleting files with <code>`unlink()`</code> is potentially dangerous. A malicious actor could use this to modify or access files they have no right to.	3	/legacy_tests/Browser/TestCase.php:53
LOW	semgrep	Detected string concatenation with a non-literal variable in a <code>util.format / console.log</code> function. If an attacker injects a format specifier in the string, it will forge the log message. Try to use constant values for the format string.	4	/js/directives/wire-model.js:23
LOW	dep_checker	Loose version constraint: 'brotli-size^4.0.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'vitest^4.1.11' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'nprogress^0.2.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'esbuild^0.14.49' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/anchor^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'alpinejs^3.17.4' — attacker could push malicious update	1	package.json

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	dep_checker	Loose version constraint: '@vue/reactivity^3.2.45' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/sort^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/resize^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/persist^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/morph^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/mask^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/intersect^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/focus^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/csp^3.17.4' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@alpinejs/collapse^3.17.4' — attacker could push malicious update	1	package.json

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.