

cocoon

github.com/xserver-inc/cocoon

Scanned 24 September 2026 at 08:10 · Report generated 29 October 2026 · PHP · 144 stars

OVERALL ASSESSMENT

High

Combined score 224 · 109 findings after filtering

Summary

This repository has a high vulnerability score of 191.0, which means there are many dangerous coding patterns that an attacker could exploit to break your application or steal data. While the code also shows some signs of malicious intent with a medium score of 32.0, most of these issues appear to be standard security mistakes rather than deliberate attacks by the developers. The main problems include untrusted input leading to Cross-site scripting (XSS) vulnerabilities and the execution of downloaded files, which are common errors in web development that can be fixed easily. Additionally, there are several instances where code directly accesses user data without proper escaping or uses unsafe functions like `dangerouslySetInnerHTML` that could cause similar XSS attacks if not handled correctly. The scan also identified some unusual patterns like using mutable GitHub Actions tags and potentially obfuscated strings, but these do not necessarily indicate a coordinated attack against the project itself.

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 (191)

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

MALICIOUS PATTERN

Medium (32)

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	3	73	0	76
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	20	20
yara	Signatures of malicious behaviour in source files	6	0	3	9
guarddog	Malicious package and install-time behaviour	3	1	0	4

67 of 109 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	Untrusted input could be used to tamper with a web page rendering, which can lead to a Cross-site scripting (XSS) vulnerability. XSS vulnerabilities occur when untrusted input executes malicious JavaScript code, leading to issues such as account compromise and sensitive information leakage. To preve	1	/lib/page-access/ analytics/ dashboard-page.php:39 0
HIGH	guarddog	Detects download-and-execute patterns: fetching a remote file then executing it [threat-process-download-exec]	3	/scripts/ setup-wpcli.js:83
HIGH	semgrep	User data flows into this manually-constructed SQL string. User data can be safely inserted into SQL strings using prepared statements or an object-relational mapper (ORM). Manually-constructed SQL strings is a possible indicator of SQL injection, which could let an attacker steal or manipulate data	1	/lib/ plugin-update-checker/ Puc/v5p6/ Plugin/ Ui.php:227
HIGH	semgrep	Executing non-constant commands. This can lead to command injection.	1	/scripts/ audit-cocoon-settings-data-contract.php:36

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	2	/plugins/ highlight- js/ highlight. all.min.js
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	3	/plugins/ lightbox2/ dist/js/ lightbox- plus- jquery.js
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/plugins/ highlight- js/ highlight. all.min.js
MEDIUM	semgrep	RegExp() called with a `className` 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-contr	1	/plugins/ scroll- hint- master/js/ scroll- hint.js:278
MEDIUM	semgrep	RegExp() called with a `_ref` 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	/webfonts/ fontawesom e5/js/ fontawesome .js:1376
MEDIUM	semgrep	Found direct access to a PHP variable without HTML escaping inside an inline PHP statement setting data from `\$_REQUEST[...]`. When untrusted input can be used to tamper with a web page rendering, it can lead to a Cross-site scripting (XSS) vulnerability. XSS vulnerabilities occur when untrusted input	44	/lib/page- func-text/ list.php:80

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	guarddog	Detects unicode homoglyphs and uncommon characters used for obfuscation [threat-runtime-obfuscation-unicode]	1	/plugins/highlight-js/highlight.all.min.js:2
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	15	/.github/workflows/blocks.yml:33
MEDIUM	semgrep	Detection of dangerouslySetInnerHTML 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 dangerouslySetInnerHTML, consider using a sanitization library such as DOMPurify to sanitize your HT	1	/blocks/src/block/button-wrap/edit.js:85
MEDIUM	semgrep	Detected possible user input going into a `path.join` or `path.resolve` 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.	8	/scripts/test-i18n.js:892
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.	3	/lib/utls.php:2062
LOW	dep_checker	Loose version constraint: 'run-script-os^1.1.6' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'stylelint-scss^6.0.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'stylelint^16.0.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'gulp^5.0.0' — attacker could push malicious update	1	package.json

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	dep_checker	Loose version constraint: '@prettier/plugin-php^0.19.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@wordpress/eslint-plugin^16.0.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@wordpress/prettier-config^2.6.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'eslint^8.57.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'gulp-plumber^1.2.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'gulp-rename^2.0.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'gulp-sass^5.1.0' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: '@playwright/test^1.62.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'jsdom^20.0.3' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'lighthouse^13.4.1' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'po2json^0.4.5' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'postcss-scss^4.0.9' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'postcss^8.5.23' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'sass^1.69.5' — attacker could push malicious update	1	package.json
LOW	dep_checker	Loose version constraint: 'stylelint-config-standard-scss^13.0.0' — attacker could push malicious update	1	package.json

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	dep_checker	Loose version constraint: 'stylelint-config-standard^36.0.0' — 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	3	/.github/ workflows/ apply- translation -fallbacks. yaml

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.