

homebrew-core

github.com/Homebrew/homebrew-core

Scanned 24 September 2026 at 08:59 · Report generated 29 October 2026 · Ruby · 15,585 stars

OVERALL ASSESSMENT

High

Combined score 196 · 132 findings after filtering

Summary

The repository Homebrew/homebrew-core has been flagged as having a high vulnerability risk of 122.0 and a high malicious-pattern risk of 74.0, which means the code contains several dangerous patterns that could be exploited by attackers or indicate intentional bad behavior. The most concerning issues include detected cryptocurrency mining activity in files like `monero.rb`, which suggests the project might be running unauthorized crypto-mining processes, and multiple hardcoded secrets such as API keys and private keys that should never be stored directly in source code. While some findings are standard insecure coding practices like using unencrypted WebSocket connections or storing bcrypt hashes in plain text, these are routine mistakes rather than signs of malicious intent. The presence of YARA rules matching cryptocurrency mining patterns and download-and-execute logic further confirms that the project actively supports or facilitates crypto-mining operations. Because of these specific risks related to crypto-mining and hardcoded credentials, this repository is not safe to use for any production environment without significant remediation efforts.

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 (122)

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

MALICIOUS PATTERN

High (74)

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
yara	Signatures of malicious behaviour in source files	73	0	5	78
semgrep	Insecure coding patterns across many languages	12	24	0	36
guarddog	Malicious package and install-time behaviour	3	10	5	18

50 of 132 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	Found a potentially user-controllable argument in the construction of a regular expressions. This may result in excessive resource consumption when applied to certain inputs, or when the user is allowed to control the match target. Avoid allowing users to specify regular expressions processed by the	1	/Formula/u/unibilium.rb:61
HIGH	guarddog	Detects cryptocurrency mining activity [threat-process-cryptomining]	3	/Formula/m/monero.rb:2
HIGH	semgrep	Generic API Key detected	1	/Formula/f/fred.rb:54
HIGH	semgrep	Generic Secret detected	5	/Formula/m/mdz.rb:44
HIGH	semgrep	Insecure WebSocket Detected. WebSocket Secure (wss) should be used for all WebSocket connections.	1	/Formula/w/wstunnel.rb:39
HIGH	semgrep	JWT token detected	1	/Formula/j/jwt-hack.rb:31
HIGH	semgrep	bcrypt hash detected	2	/Formula/lib/libxcrypt.rb:58
HIGH	semgrep	Private Key detected. This is a sensitive credential and should not be hardcoded here. Instead, store this in a separate, private file.	1	/Formula/t/trojan-go.rb:98

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_crypto_miner — Cryptocurrency mining pool connection patterns	3	/Formula/x/xmrig.rb
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	3	/Formula/o/openbao.rb
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	67	/Formula/z/zsh-autopair.rb
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.	10	/Formula/g/gitup.rb:46
MEDIUM	semgrep	Detected non-static command inside PTY.spawn. Audit the input to 'PTY.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.	1	/Formula/n/nak.rb:39
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	/Formula/g/grokmirror.rb:62
MEDIUM	guarddog	Detects URLs to suspicious domains often used for exfiltration or C2 [threat-network-exfiltration]	5	/Formula/j/jimtbl.rb:3
MEDIUM	guarddog	Detects URLs to URL shorteners, file sharing, and suspicious services [threat-network-outbound-shady-links]	5	/Formula/c/cidr2range.rb:3
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.	11	/Formula/p/portable-ruby.rb:161

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	guarddog	Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]	5	/Formula/c/container-structure-test.rb:48
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	5	/Formula/p/pushpin.rb

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.