

flatpress

github.com/flatpressblog/flatpress

Scanned 24 September 2026 at 07:21 · Report generated 29 October 2026 · PHP · 215 stars

OVERALL ASSESSMENT

Critical

Combined score 495 · 225 findings after filtering

Summary

This repository has been flagged as having a critical vulnerability score of 494.0, which means there are serious security issues in the code that an attacker could easily exploit to break into your system or steal data. The scan found many instances where untrusted user input is being used directly without proper protection, such as when creating links or executing commands based on what a visitor types in a form. While the code does not appear to be intentionally malicious, these patterns are typical of insecure coding practices that developers often use but fail to secure properly. Because of these dangerous flaws, it is not safe to use this repository for any production project without first fixing the identified vulnerabilities.

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

Critical (494)

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	65	154	0	219

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
yara	Signatures of malicious behaviour in source files	6	0	0	6

50 of 225 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	3	/fp-plugins/tag/inc/admin.php:619
HIGH	semgrep	The application might dynamically evaluate untrusted input, which can lead to a code injection vulnerability. An attacker can execute arbitrary code, potentially gaining complete control of the system. To prevent this vulnerability, avoid executing code containing user input. If this is unavoidable,	2	/admin/panels/widgets/admin.widgets.raw.php:49
HIGH	semgrep	Detected user input going into a php include or require command, which can lead to path traversal and sensitive data being exposed. These commands can also lead to code execution. Instead, allowlist files that the user can access or rigorously validate user input.	2	/setup/main.php:87
HIGH	semgrep	The 'phpinfo' function may reveal sensitive information about your environment.	1	/admin/panels/maintain/admin.maintain.php:672
HIGH	semgrep	Executing non-constant commands. This can lead to command injection.	49	/fp-plugins/tag/inc/tagdb.php:267

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Evaluating non-constant commands. This can lead to command injection.	8	/fp-includes/smarty-5.8.4/src/FunctionHandler/Math.php:124
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	1	/fp-includes/core/core.utils.php
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	5	/fp-plugins/jquery/res/jqueryui/1.14.2/jquery-ui.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	5	/fp-plugins/photoswipe/res/jquery-3.7.1/jquery-3.7.1.js:1651
MEDIUM	semgrep	RegExp() called with a `types` 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-controlle	1	/fp-plugins/jquery/res/jquery/4.0.0/jquery-4.0.0.js:4798
MEDIUM	semgrep	<- A new object is created where the class name is based on user input. This could lead to remote code execution, as it allows to instantiate any class in the application.	1	/admin/main.php:67

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
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	4	/fp-plugins/seometataginfo/plugin.seometataginfo.php:251
MEDIUM	semgrep	The application builds a file path from potentially untrusted data, which can lead to a path traversal vulnerability. An attacker can manipulate the file path which the application uses to access files. If the application does not validate user input and sanitize file paths, sensitive files such as	2	/setup/main.php:87
MEDIUM	semgrep	Callable based on user input risks remote code execution.	1	/admin/includes/panels.prototypes.php:173
MEDIUM	semgrep	User data flows into the host portion of this manually-constructed URL. This could allow an attacker to send data to their own server, potentially exposing sensitive data such as cookies or authorization information sent with this request. They could also probe internal servers or other resources through	1	/fp-plugins/commentcenter/inc/admin.php:204
MEDIUM	semgrep	File name based on user input risks server-side request forgery.	17	/fp-plugins/mediamanager/panels/panel.mediamanager.file.php:263
MEDIUM	semgrep	Make sure comparisons involving md5 values are strict (use `===` not `==`) to avoid type juggling issues	1	/fp-plugins/bbcode/plugin.bbcode.php:517
MEDIUM	semgrep	Access-Control-Allow-Origin response header is set to "*". This will disable CORS Same Origin Policy restrictions.	1	/fp-plugins/feed/res/feed.css.php:3

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	The function <code>`openssl_decrypt`</code> returns either a string of the decrypted data on success or <code>`false`</code> on failure. If the failure case is not handled, this could lead to undefined behavior in your application. Please handle the case where <code>`openssl_decrypt`</code> returns <code>`false`</code> .	2	/fp-plugins/mastodon/plugin.mastodon.php:221
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. <code>`uses: ac</code>	6	/.github/workflows/phpstan.yml:22
MEDIUM	semgrep	Possibility of prototype polluting function detected. By adding or modifying attributes of an object prototype, it is possible to create attributes that exist on every object, or replace critical attributes with malicious ones. This can be problematic if the software depends on existence or non-exis	22	/fp-plugins/jquery/res/jquery/4.0.0/jquery-4.0.0.js:1955
MEDIUM	semgrep	Calling <code>`unserialize()`</code> with user input in the pattern can lead to arbitrary code execution. Consider using JSON or structured data approaches (e.g. Google Protocol Buffers).	8	/fp-plugins/lastcomments/plugin.lastcomments.php:49
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.	82	/admin/panels/maintain/admin.maintain.php:586

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.