

# newznab-tmux

github.com/NNTmux/newznab-tmux

Scanned 24 September 2026 at 09:32 · Report generated 29 October 2026 · PHP · 295 stars

## OVERALL ASSESSMENT

# High

Combined score 172 · 96 findings after filtering

## Summary

This repository has a high vulnerability score of 140.0, which means there are many dangerous coding patterns that an attacker could easily exploit to break into your system. While the code does contain some signs of malicious intent, most of these issues appear to be standard mistakes in how developers write software rather than intentional attacks. The scan found several critical problems, including SQL injection vulnerabilities where user data is directly inserted into database queries without proper protection. It also flagged access to sensitive files like credentials and configuration keys, as well as commands that could be used to execute arbitrary code if the input is not handled correctly. Additionally, there are concerns about how secrets are stored and permissions on files that might grant too much access to unintended users. The presence of YARA rules indicates that the codebase may contain malware evasion techniques or even attempts to download and run malicious scripts. Despite these serious findings, the developer should still proceed with caution and apply fixes for the high-risk issues before deploying anything to production.

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 (140)

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

### MALICIOUS PATTERN

## Medium (33)

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     |
|--------------------|---------------------------------------------------|------|--------|-----|-----------|
| <b>semgrep</b>     | Insecure coding patterns across many languages    | 17   | 28     | 0   | <b>45</b> |
| <b>dep_checker</b> | Vulnerable, unpinned or typosquatted dependencies | 0    | 0      | 20  | <b>20</b> |
| <b>yara</b>        | Signatures of malicious behaviour in source files | 13   | 0      | 4   | <b>17</b> |
| <b>bandit</b>      | Insecure coding patterns in Python source         | 0    | 1      | 9   | <b>10</b> |
| <b>guarddog</b>    | Malicious package and install-time behaviour      | 3    | 0      | 1   | <b>4</b>  |

34 of 96 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                                             |
|-------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|--------------------------------------------------------------|
| <b>HIGH</b> | semgrep  | Untrusted input might be used to build a database query, which can lead to a SQL injection vulnerability. An attacker can execute malicious SQL statements and gain unauthorized access to sensitive data, modify, delete data, or execute arbitrary system commands. To prevent this vulnerability, use pre | 2    | /app/<br>Services/<br>MusicService<br>.php:192               |
| <b>HIGH</b> | guarddog | Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]                                                                                                                                                                                                                      | 3    | /deploy/<br>baremetal/<br>host.py:280                        |
| <b>HIGH</b> | 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 | 4    | /app/<br>Services/<br>MusicService<br>.php:187               |
| <b>HIGH</b> | semgrep  | Generic API Key detected                                                                                                                                                                                                                                                                                     | 1    | /.env.examp<br>le:271                                        |
| <b>HIGH</b> | semgrep  | Executing non-constant commands. This can lead to command injection.                                                                                                                                                                                                                                         | 9    | /app/<br>Console/<br>Commands/<br>NntmuxESRei<br>ndex.php:42 |

| SEVERITY | DETECTOR | FINDING                                                                                                                                                                                                                                                                                                | SEEN | EXAMPLE LOCATION                                        |
|----------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|---------------------------------------------------------|
| HIGH     | semgrep  | By not specifying a USER, a program in the container may run as 'root'. This is a security hazard. If an attacker can control a process running as root, they may have control over the container. Ensure that the last USER in a Dockerfile is a USER other than 'root'.                              | 1    | /docker/8.5/Dockerfile:104                              |
| HIGH     | yara     | YARA rule matched: detect_anti_analysis — Checks for a debugger, VM, or sandbox before continuing - a common malware evasion technique, essentially never present in ordinary application code                                                                                                         | 1    | /scripts/benchmark-yenc.php                             |
| HIGH     | yara     | YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed                                                                                                                                                                                                                      | 2    | /public/js/filament/forms/components/file-upload.js     |
| 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    | /scripts/build-rapidync.sh                              |
| 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    | /public/js/filament/forms/components/markdown-editor.js |
| HIGH     | yara     | YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this                                                                                                                                                     | 4    | /infra/terraform/aws/cloud-init.yaml.tftpl              |

| SEVERITY      | DETECTOR | FINDING                                                                                                                                                                                                                                                                                                      | SEEN | EXAMPLE LOCATION                                                               |
|---------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|--------------------------------------------------------------------------------|
| <b>MEDIUM</b> | semgrep  | RegExp() called with a `query` 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    | /resources/<br>js/alpine/<br>components/<br>search-<br>autocomplete<br>.js:138 |
| <b>MEDIUM</b> | semgrep  | Ensure EBS Volume is encrypted at rest using KMS CMKs. CMKs gives you control over the encryption key in terms of access and rotation.                                                                                                                                                                       | 1    | /infra/<br>terraform/<br>aws/<br>main.tf:100                                   |
| <b>MEDIUM</b> | semgrep  | By default, AWS SecretManager secrets are encrypted using AWS-managed keys. However, for added security, it's recommended to configure your own AWS KMS encryption key to protect your secrets in the Secret Manager. You can either create a new aws_kms_key resource or use the ARN of an existing key in  | 1    | /infra/<br>terraform/<br>aws/<br>main.tf:123                                   |
| <b>MEDIUM</b> | semgrep  | These permissions `0o700` are widely permissive and grant access to more people than may be necessary. A good default is `0o644` which gives read and write access to yourself and read access to everyone else.                                                                                             | 1    | /deploy/<br>baremetal/<br>host.py:773                                          |
| <b>MEDIUM</b> | semgrep  | EC2 instances should not have a public IP address attached in order to block public access to the instances. To fix this, set your `associate_public_ip_address` to `"false"`.                                                                                                                               | 1    | /infra/<br>terraform/<br>aws/<br>main.tf:169                                   |
| <b>MEDIUM</b> | semgrep  | This Dependabot configuration does not set a cooldown period. Newly published packages can be malicious or unstable. Add a `cooldown` block with `default-days: 7` to each `package-ecosystem` entry under `updates` to wait 7 days before proposing updates to newly published package versions. Reference: | 1    | /.github/<br>dependabot.<br>yaml:8                                             |
| <b>MEDIUM</b> | semgrep  | File name based on user input risks server-side request forgery.                                                                                                                                                                                                                                             | 1    | /<br>server.php:1<br>5                                                         |

| SEVERITY      | DETECTOR | FINDING                                                                                                                                                                                                                                                                                                      | SEEN | EXAMPLE LOCATION                                          |
|---------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|-----------------------------------------------------------|
| <b>MEDIUM</b> | 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 | 16   | /.github/workflows/terraform.yml:132                      |
| <b>MEDIUM</b> | 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 | 2    | /public/js/filament/widgets/components/chart.js:1         |
| <b>MEDIUM</b> | 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).                                                                                                                               | 2    | /app/Services/IRCScraper.php:73                           |
| <b>MEDIUM</b> | bandit   | Probable insecure usage of temp file/directory.                                                                                                                                                                                                                                                              | 1    | /deploy/baremetal/host.py:772                             |
| <b>MEDIUM</b> | 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.                                                                                                                                             | 1    | /scripts/benchmark-yenc.php:145                           |
| <b>LOW</b>    | guarddog | Detects heavy obfuscation techniques commonly used by malware [threat-runtime-obfuscation]                                                                                                                                                                                                                   | 1    | /public/js/filament/forms/components/markdown-editor.js:1 |
| <b>LOW</b>    | bandit   | Consider possible security implications associated with the subprocess module.                                                                                                                                                                                                                               | 2    | /deploy/baremetal/host.py:11                              |
| <b>LOW</b>    | bandit   | Starting a process with a partial executable path                                                                                                                                                                                                                                                            | 2    | /deploy/baremetal/deploy.py:193                           |
| <b>LOW</b>    | bandit   | subprocess call - check for execution of untrusted input.                                                                                                                                                                                                                                                    | 5    | /deploy/baremetal/host.py:373                             |

| SEVERITY | DETECTOR    | FINDING                                                                                              | SEEN | EXAMPLE LOCATION |
|----------|-------------|------------------------------------------------------------------------------------------------------|------|------------------|
| LOW      | dep_checker | Loose version constraint: 'lodash^4.18.1' — attacker could push malicious update                     | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'chart.js^4.5.1' — attacker could push malicious update                    | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'date-fns^4.4.0' — attacker could push malicious update                    | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: '@alpinejs/csp^3.17.3' — attacker could push malicious update              | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'feather-icons^4.29.2' — attacker could push malicious update              | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: '@fortawesome/free^7.3.1' — attacker could push malicious update           | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: '@melloware/coloris^0.25.0' — attacker could push malicious update         | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'laravel-echo^2.5.0' — attacker could push malicious update                | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'laravel-vite-plugin^3.2' — attacker could push malicious update           | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: '@tailwindcss/vite^4.3.3' — attacker could push malicious update           | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'nested-sort^5.2.0' — attacker could push malicious update                 | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'npm-check-updates^23.1.0' — attacker could push malicious update          | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: '@simonwep/pickr^1.10.2' — attacker could push malicious update            | 1    | package.json     |
| LOW      | dep_checker | Loose version constraint: 'prettier-plugin-tailwindcss^0.7.4' — attacker could push malicious update | 1    | package.json     |

| SEVERITY | DETECTOR    | FINDING                                                                                                                                                     | SEEN | EXAMPLE LOCATION       |
|----------|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|------|------------------------|
| LOW      | dep_checker | Loose version constraint: '@simplewebauthn/browser^14.0.0' — attacker could push malicious update                                                           | 1    | package.json           |
| LOW      | dep_checker | Loose version constraint: 'tailwindcss^4.3.3' — attacker could push malicious update                                                                        | 1    | package.json           |
| LOW      | dep_checker | Loose version constraint: 'vite^8.3' — attacker could push malicious update                                                                                 | 1    | package.json           |
| LOW      | dep_checker | Loose version constraint: 'vue^3.5.43' — attacker could push malicious update                                                                               | 1    | package.json           |
| LOW      | dep_checker | Loose version constraint: '@tailwindcss/forms^0.5.11' — attacker could push malicious update                                                                | 1    | package.json           |
| LOW      | dep_checker | Loose version constraint: 'vuedraggable^4.1.0' — attacker could push malicious update                                                                       | 1    | package.json           |
| 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 | 4    | /docker/8.5/Dockerfile |

## 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.