

neutree

github.com/neutree-ai/neutree

Scanned 24 September 2026 at 09:03 · Report generated 29 October 2026 · Go · 140 stars

OVERALL ASSESSMENT

Critical

Combined score 345 · 141 findings after filtering

Summary

This repository has two distinct security issues that require immediate attention before it can be considered safe for production use. The most critical finding is an insecure gRPC connection created by `insecure.NewCredentials()`, which means any attacker could intercept or tamper with data sent between your internal observability tools and Kubernetes pods without encryption. Additionally, the code uses a secret-inheritance pattern that grants too much access to reusable workflows, and there are several other high-priority vulnerabilities including unverified command execution, overly permissive CORS policies, and potential supply-chain attacks via mutable tags in GitHub Actions. While some findings appear to be legitimate application behavior rather than malicious intent, the combination of these insecure coding patterns and configuration errors creates a significant risk that prevents the repository from being used safely without further hardening.

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 (344)

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

MALICIOUS PATTERN

Low (1)

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	36	79	0	115
bandit	Insecure coding patterns in Python source	0	4	13	17
yara	Signatures of malicious behaviour in source files	2	0	5	7
guarddog	Malicious package and install-time behaviour	0	0	1	1
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	1	1

109 of 141 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 an insecure gRPC connection using 'insecure.NewCredentials()'. This creates a connection without encryption to a gRPC server. A malicious attacker could tamper with the gRPC message, which could compromise the machine. Instead, establish a secure connection with an SSL certificate using the 'g	1	/internal/observability/ neutreeometrics/ kubernetes/ podresources_ client.go:40
HIGH	semgrep	This workflow uses `secrets: inherit` to pass all of the calling workflow's secrets to a reusable workflow. This violates the principle of least privilege because the called workflow receives access to every secret in the repository, not just the ones it needs. If the called workflow is compromised	3	/.github/ workflows/ release- engine-image- llama- cpp.yaml:93
HIGH	semgrep	Using variable interpolation `\${{...}}` with `github` context data in a `run:` step could allow an attacker to inject their own code into the runner. This would allow them to steal secrets and code. `github` context data can have arbitrary user input and should be treated as untrusted. Instead, use	29	/.github/ workflows/ release- neutree- runtime.yaml: 49

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Detected non-static command inside Command. Audit the input to 'exec.Command'. 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.	2	/internal/observability/ neutreeometrics/adapters/nvidia/nvidia_processes.go:51
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	/cluster-image-builder/Dockerfile:84
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	/.github/workflows/cleanup-nightly.yaml
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/cluster-image-builder/Dockerfile
MEDIUM	semgrep	ReverseProxy can remove headers added by Director. Consider using ReverseProxy.Rewrite instead of ReverseProxy.Director.	1	/internal/routes/proxies/proxies.go:55
MEDIUM	semgrep	`MinVersion` is missing from this TLS configuration. By default, as of Go 1.22, TLS 1.2 is currently used as the minimum. General purpose web applications should default to TLS 1.3 with all other protocols disabled. Only where it is known that a web server must support legacy clients with unsupported	5	/pkg/client/client.go:48
MEDIUM	semgrep	CORS policy allows any origin (using wildcard '*'). This is insecure and should be avoided.	4	/cluster-image-builder/serve/llama_cpp/v0_3_7/app.py:219

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
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>	53	<code>/.github/workflows/release-engine-image-sglang.yaml:83</code>
MEDIUM	semgrep	When working with web applications that involve rendering user-generated content, it's important to properly escape any HTML content to prevent Cross-Site Scripting (XSS) attacks. In Go, the <code>`text/template`</code> package does not automatically escape HTML content, which can leave your application vuln	4	<code>/internal/cluster/component/metrics/vmagent_config.go:5</code>
MEDIUM	semgrep	Deserializing into <code>`interface{}`</code> allows arbitrary data structures and types, which can lead to security vulnerabilities (CWE-502). Use a concrete struct type instead.	5	<code>/internal/orchestrator/kubernetes_orchestrator_resource.go:216</code>
MEDIUM	semgrep	Checks for disabling of TLS/SSL certificate verification. This should only be used for debugging purposes because it leads to vulnerability to MTM attacks.	4	<code>/pkg/client/client.go:48</code>
MEDIUM	bandit	Use of weak MD5 hash for security. Consider <code>usedforsecurity=False</code>	2	<code>/cluster-image-builder/serve/_replica_scheduler/chwbl_scheduler.py:180</code>
MEDIUM	semgrep	Untrusted user input in <code>`importlib.import_module()`</code> function allows an attacker to load arbitrary code. Avoid dynamic values in <code>`importlib.import_module()`</code> or use a whitelist to prevent running untrusted code.	1	<code>/cluster-image-builder/start.py:34</code>
MEDIUM	semgrep	Detected a possible denial-of-service via a zip bomb attack. By limiting the max bytes read, you can mitigate this attack. <code>`io.CopyN()`</code> can specify a size.	2	<code>/internal/model_registry/bentoml/bentoml.go:703</code>

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	bandit	Probable insecure usage of temp file/directory.	2	/cluster-image-builder/serve/_metrics/prometheus_multiproc.py:36
LOW	bandit	Possible hardcoded password: 'NEUTREE_DL_TOKEN'	1	/python/neutree/downloader/utils.py:68
LOW	bandit	Possible hardcoded password: 'MODELSCOPE_API_TOKEN'	1	/python/neutree/downloader/utils.py:69
LOW	bandit	Possible hardcoded password: 'HF_TOKEN'	1	/python/neutree/downloader/utils.py:70
LOW	guarddog	Detects reading of environment variables (credential access, often contains secrets) [threat-runtime-environment-read]	1	/cmd/neutree-cli/app/cmd/global/global.go:32
LOW	bandit	Possible hardcoded password: "	1	/python/neutree/downloader/utils.py:137
LOW	bandit	Try, Except, Continue detected.	1	/python/neutree/downloader/progress.py:22
LOW	bandit	Consider possible security implications associated with the subprocess module.	3	/cluster-image-builder/start.py:2
LOW	bandit	Starting a process with a partial executable path	2	/cluster-image-builder/accelerator/gpu.py:15

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	subprocess call - check for execution of untrusted input.	3	/cluster-image-builder/accelerator/amd_gpu.py:11
LOW	dep_checker	Loose version constraint: 'click>=7.0, <8.3.0' — attacker could push malicious update	1	requirements.txt:3
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	/cluster-image-builder/Dockerfile.rocm

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.