

clawmetry

github.com/vivekchand/clawmetry

Scanned 07 October 2026 at 04:47 · Report generated 29 October 2026 · Python · 424 stars

OVERALL ASSESSMENT

Critical

Combined score 902 · 1,466 findings after filtering

Summary

This repository shows severe security issues that are likely caused by malicious intent rather than just poor coding habits, as indicated by the Critical scores of 584.0 and 317.0 from automated tools like Bandit and YARA. The most concerning findings involve dangerous patterns such as recursive file deletion, dynamic code loading for execution, and cryptocurrency mining activity, which pose immediate threats to system stability and data integrity. Additionally, there are multiple hardcoded passwords and messaging platform tokens that could be exploited to steal sensitive information or bypass security controls. While some issues like URL schemes and XML parsing vulnerabilities appear less critical, the presence of these malicious patterns suggests the codebase is actively compromised rather than just containing standard bugs.

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 (584)

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

MALICIOUS PATTERN

Critical (317)

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
bandit	Insecure coding patterns in Python source	0	309	961	1,270

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
yara	Signatures of malicious behaviour in source files	63	0	77	140
guarddog	Malicious package and install-time behaviour	23	15	10	48
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	8	8

73 of 1,466 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	guarddog	Detects suspicious autostart persistence mechanisms [threat-filesystem-autostart]	3	/ dashboard.py:1758
HIGH	guarddog	Detects dynamic code loading: downloading and importing/executing code at runtime [threat-runtime-dynamic-loader]	3	/clawmetry/sync.py:931
HIGH	guarddog	Detects destructive operations (recursive deletion, wiping) [threat-filesystem-destruction]	2	/clawmetry/framework_map.py:197
HIGH	guarddog	Detects URLs to suspicious domains often used for exfiltration or C2 [threat-network-exfiltration]	5	/routes/alerts.py:1166
HIGH	guarddog	Detects system info collection combined with network exfiltration (hostname/user in HTTP requests) [threat-network-exfil-sysinfo]	3	/ dashboard.py:3128
HIGH	guarddog	Detects hardcoded messaging platform tokens/webhooks used for data exfiltration [threat-network-exfil-messenger]	3	/clawmetry/incident_alerts.py:360
HIGH	guarddog	Detects cryptocurrency mining activity [threat-process-cryptomining]	3	/ dashboard.py:10065
HIGH	guarddog	Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]	5	/clawmetry/detector_behaviour.py:186
HIGH	guarddog	Detects download-and-execute patterns: fetching a remote file then executing it [threat-process-download-exec]	3	/ dashboard.py:3128

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_crypto_miner — Cryptocurrency mining pool connection patterns	1	/dashboard.py
HIGH	yara	YARA rule matched: detect_credential_harvester — References credential/secret material by name AND a deliberate outbound-send API in the same file - narrower than 'reads any env var', which large legitimate codebases do constantly regardless of intent. Still not proof the two are connected (YARA c	10	/install.sh
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	39	/clawmetry/detector_behaviour.py
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	2	/clawmetry/static/v2/dist/assets/index-CrUi8oj9.js
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	11	/install-clawmetry.sh
MEDIUM	guarddog	Detects extensive system/network enumeration activities [threat-runtime-enumeration]	3	/dashboard.py:1800
MEDIUM	guarddog	Detects URLs to URL shorteners, file sharing, and suspicious services [threat-network-outbound-shady-links]	5	/clawmetry/briefs.py:364
MEDIUM	bandit	Audit url open for permitted schemes. Allowing use of file:/ or custom schemes is often unexpected.	116	/clawmetry/license.py:901
MEDIUM	bandit	Possible SQL injection vector through string-based query construction.	191	/clawmetry/local_store.py:2712

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	bandit	Probable insecure usage of temp file/directory.	2	/dashboard.py:12595
LOW	bandit	Using escape to parse untrusted XML data is known to be vulnerable to XML attacks. Replace escape with the equivalent defusedxml package, or make sure defusedxml.defuse_stdlib() is called.	3	/clawmetry/cli.py:1790
LOW	bandit	Possible hardcoded password: '[CMPII:'	1	/clawmetry/assessment_privacy.py:29
LOW	bandit	Possible hardcoded password: 'tokens consumed by this sub-agent'	1	/clawmetry/dives_prompt.py:178
LOW	bandit	Possible hardcoded password: 'total tokens consumed on this day'	1	/clawmetry/dives_prompt.py:118
LOW	bandit	Possible hardcoded password: 'guard.last_pass_ms'	1	/clawmetry/guard_checks.py:10
LOW	bandit	Possible hardcoded password: 'count_over_threshold'	1	/clawmetry/alert_evaluator.py:74
LOW	bandit	Possible hardcoded password: 'first_pass_rate_below'	1	/clawmetry/alert_evaluator.py:102
LOW	bandit	Possible hardcoded password: 'CLAW1'	1	/clawmetry/license.py:73
LOW	bandit	Possible hardcoded password: 'ok'	1	/scripts/redteam/audit.py:399
LOW	bandit	Possible hardcoded password: 'ok '	1	/scripts/redteam/audit.py:471
LOW	bandit	Possible hardcoded password: '2.0'	2	/routes/usage.py:2476

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Possible hardcoded password: '[OK]'	1	/clawmetry/ doctor.py:26
LOW	bandit	Possible hardcoded password: 'token_spike'	1	/routes/ alerts.py:350
LOW	bandit	Possible hardcoded password: 'sum of token counts on deduped call events in the window'	1	/routes/ usage.py:357
LOW	guarddog	Detects active collection of system information (hostname, platform, architecture, user) [threat-runtime-system-info]	3	/clawmetry/ cli.py:414
LOW	guarddog	Detects reading of environment variables (credential access, often contains secrets) [threat-runtime-environment-read]	1	/.github/ scripts/ visual- diff.mjs:49
LOW	guarddog	Detects memory scraping and credential dumping from process memory [threat-process-memory]	3	/clawmetry/ process_co ntrol.py:815
LOW	bandit	Possible hardcoded password: '[WITHHELD:secret]'	1	/clawmetry/ assessment_ privacy.py:28
LOW	bandit	Possible hardcoded password: 'recorded token usage; verify interval, cumulative and cache basis before summing'	1	/clawmetry/ dives_prom pt.py:79
LOW	guarddog	Detects fully silent process execution (suppressing all output channels) [threat-process-spawn-silent]	3	/clawmetry/ process_co ntrol.py:893
LOW	bandit	Possible hardcoded password: 'PASS'	2	/scripts/ accuracy_h arness/ keystone_e2 e.py:79
LOW	bandit	Possible hardcoded password: '0'	2	/clawmetry/ sync.py:13967
LOW	bandit	Possible hardcoded password: 'pass'	1	/clawmetry/ repo_readi ness.py:67

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Using xml.etree.ElementTree to parse untrusted XML data is known to be vulnerable to XML attacks. Replace xml.etree.ElementTree with the equivalent defusedxml package, or make sure defusedxml.defuse_stdlib() is called.	1	/scripts/auto_quarantine.py:41
LOW	bandit	Possible hardcoded password: "	15	/dashboard.py:2702
LOW	bandit	Possible hardcoded password: 'None'	7	/clawmetry/adapters/nemo.py:497
LOW	bandit	Starting a process without a shell.	2	/clawmetry/sync.py:25467
LOW	bandit	Try, Except, Continue detected.	232	/clawmetry/capability_gaps.py:321
LOW	bandit	Consider possible security implications associated with the subprocess module.	140	/clawmetry/process_control.py:1316
LOW	bandit	Starting a process with a partial executable path	202	/scripts/pr_provenance_demo.py:39
LOW	dep_checker	Loose version constraint: 'certifi>=2024.7.4 # CA bundle: the trust-store fallback for 3.8/3.9 and for any interpreter with no CA store. Floor is the first release that drops the distrusted GLOBALTRUST root (CVE-2024-39689); same requires_python (' — attacker could push malicious update	1	requirements.txt:4
LOW	bandit	subprocess call - check for execution of untrusted input.	316	/clawmetry/daemon_registration.py:173
LOW	dep_checker	Loose version constraint: 'zstandard>=0.25.0; python_version < "3.14" # decodes OpenClaw 2026.9.x zstd transcript events (stdlib from 3.14)' — attacker could push malicious update	1	requirements.txt:5

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	dep_checker	Loose version constraint: 'truststore>=0.10.4; python_version' — attacker could push malicious update	1	requirements.txt:3
LOW	dep_checker	Loose version constraint: 'duckdb>=1.5.6 # local event store (epic #964 phase 1)' — attacker could push malicious update	1	requirements.txt:2
LOW	dep_checker	Loose version constraint: 'requests>=2.34.2' — attacker could push malicious update	1	requirements.txt:10
LOW	dep_checker	Loose version constraint: 'pytest>=9.1.1' — attacker could push malicious update	1	requirements.txt:8
LOW	dep_checker	Loose version constraint: 'pytest-playwright>=0.9.0' — attacker could push malicious update	1	requirements.txt:9
LOW	dep_checker	Loose version constraint: 'flask>=3.1.3,<4' — attacker could push malicious update	1	requirements.txt:1
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	77	/scripts/verify_runtime_filtering.py
LOW	bandit	Use of assert detected. The enclosed code will be removed when compiling to optimised byte code.	22	/desktop/smoke_windows_cleanup.py:72

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 07 October 2026.