

maldet

github.com/Skidotus/maldet

Scanned 29 October 2026 at 07:09 · Report generated 29 October 2026 · Python · 1 stars

OVERALL ASSESSMENT

High

Combined score 132 · 70 findings after filtering

Summary

20 findings likely need attention, 39 look like noise. Most notable: CVE found: CVE-2026-59951 in mcp==1.29.0 — ### Summary In affected versions, a Streamab... (99% confidence).

Generated from finding counts. A plain-English summary was not available for this scan.

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 (114)

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

MALICIOUS PATTERN

Medium (17)

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	7	25	32
yara	Signatures of malicious behaviour in source files	16	0	6	22
semgrep	Insecure coding patterns across many languages	6	4	0	10
dep_checker	Vulnerable, unpinned or typosquatted dependencies	6	0	0	6

20 of 70 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	dep_checker	CVE found: CVE-2026-59951 in mcp==1.29.0 — ### Summary In affected versions, a Streamable HTTP server in stateful mode (the default) never recli	1	requirements.txt
HIGH	dep_checker	CVE found: CVE-2026-116686 in mcp==1.29.0 — ### Summary In affected versions, when a connected MCP server declared an `outputSchema` for a tool,	1	requirements.txt
HIGH	dep_checker	CVE found: CVE-2026-114603 in opentelemetry-api==1.37.0 — ## Summary `TraceState.from_header` splits the inbound W3C `tracestate` header with a regular expres	1	requirements.txt
HIGH	dep_checker	CVE found: CVE-2026-116684 in mcp==1.29.0 — ### Summary In affected versions, the SDK's server-side bearer authentication had no way to check th	1	requirements.txt
HIGH	dep_checker	CVE found: CVE-2026-116685 in mcp==1.29.0 — ### Summary In affected versions, the SDK's Streamable HTTP and SSE client transports followed HTTP	1	requirements.txt
HIGH	dep_checker	CVE found: CVE-2026-114613 in msgpack==1.2.1 — ### Impact The msgpack-python C extension (`msgpack._msgpack`) leaks a decoded map value when dese	1	requirements.txt
HIGH	semgrep	Avoiding SQL string concatenation: untrusted input concatenated with raw SQL query can result in SQL Injection. In order to execute raw query safely, prepared statement should be used. SQLAlchemy provides TextualSQL to easily used prepared statement with named parameters. For complex SQL composition	6	/evaluate_summaries.py:210
HIGH	yara	YARA rule matched: detect_reverse_shell — Reverse shell command patterns	1	/rules.yar

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_crypto_miner — Cryptocurrency mining pool connection patterns	1	/rules.yar
HIGH	yara	YARA rule matched: detect_process_injection — Windows process-injection API sequence	1	/rules.yar
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	1	/rules.yar
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	/rules.yar
HIGH	yara	YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed	1	/rules.yar
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	8	/design/font-options.html
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	1	/rules.yar
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/rules.yar

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	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	/ scanner.py :37
MEDIUM	semgrep	Hardcoded password is used as a default argument to 'extract_archives'. This could be dangerous if a real password is not supplied.	1	/ scanner.py :124
MEDIUM	semgrep	Detected possible formatted SQL query. Use parameterized queries instead.	2	/ evaluate_summaries.py:210
MEDIUM	bandit	Possible SQL injection vector through string-based query construction.	7	/db/export_se ed.py:96
LOW	bandit	Possible hardcoded password: 'your_github_token_here'	1	/ config.ex ample.py:4
LOW	bandit	Possible hardcoded password: 'infected'	2	/ scanner.py :124
LOW	bandit	Possible hardcoded password: 'your_db_password'	1	/ config.ex ample.py:20
LOW	bandit	Consider possible security implications associated with the subprocess module.	3	/ dep_check er.py:5
LOW	bandit	Starting a process with a partial executable path	8	/ dep_check er.py:320
LOW	bandit	subprocess call - check for execution of untrusted input.	10	/ dep_check er.py:320
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	6	/rules.yar

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 29 October 2026.