

CudaText

github.com/Alexey-T/CudaText

Scanned 24 September 2026 at 09:23 · Report generated 29 October 2026 · Python · 3,201 stars

OVERALL ASSESSMENT

Medium

Combined score 79 · 182 findings after filtering

Summary

This repository has a medium vulnerability score of 67, which means there are several insecure coding patterns that an attacker could potentially exploit if the code is used in a real-world application. The most serious issues involve using shell commands and subprocess calls with `shell=True`, which can lead to command injection attacks, as well as the use of weak hash functions like MD5 and SHA1 that should be replaced for better security. While there are also signs of potential malicious intent detected by specific rules, these appear to be standard obfuscation techniques or legitimate code modifications rather than intentional malware, so they do not significantly change the overall risk profile. The scan identified 182 total findings, but only a small number of high-priority items like hardcoded passwords and dangerous YARA rule matches were flagged as critical threats. Overall, while the code contains known security risks that need to be addressed through refactoring, it does not appear to contain active malicious code or hidden backdoors based on this analysis.

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

Medium (67)

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

MALICIOUS PATTERN

Low (12)

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	10	18	137	165
yara	Signatures of malicious behaviour in source files	4	0	9	13
guarddog	Malicious package and install-time behaviour	0	4	0	4

7 of 182 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	bandit	Starting a process with a shell, possible injection detected, security issue.	4	/app/py/cuda_project_man/__init__.py:605
HIGH	bandit	subprocess call with shell=True identified, security issue.	6	/app/py/cudax_lib.py:676
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	2	/app/py/cuda_multi_installer/__init__.py
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	2	/app/cudatext.app/Contents/Resources/data/lexlib/Bashscript.lcf
MEDIUM	guarddog	Detects unicode homoglyphs and uncommon characters used for obfuscation [threat-runtime-obfuscation-unicode]	1	/app/cudatext.app/Contents/Resources/py/cuda_options_editor/cd_opts_dlg.py:1360

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	guarddog	Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]	3	/app/ cudatext.app/ Contents/ Resources/py/ cuda_lexer_detector/ lexertypes.py:4
MEDIUM	bandit	Use of weak SHA1 hash for security. Consider usedforsecurity=False	4	/app/py/sys/ requests/ auth.py:205
MEDIUM	bandit	Use of weak MD5 hash for security. Consider usedforsecurity=False	2	/app/ cudatext.app/ Contents/ Resources/py/sys/ requests/ auth.py:148
MEDIUM	bandit	Use of possibly insecure function - consider using safer ast.literal_eval.	4	/app/py/ cudax_lib.py:320
MEDIUM	bandit	Use of exec detected.	4	/app/py/ cudatext_reset_plugins.py:12
MEDIUM	bandit	Pickle and modules that wrap it can be unsafe when used to deserialize untrusted data, possible security issue.	3	/app/ cudatext.app/ Contents/ Resources/py/ cuda_options_editor/ cd_opts_dlg.py:2143
MEDIUM	bandit	Call to requests without timeout	1	/tools/ download_addons.py:65
LOW	bandit	Possible hardcoded password: '-'	1	/app/py/ cuda_testing_perf/ __init__.py:460
LOW	bandit	Possible hardcoded password: '0'	2	/app/py/ cuda_testing_perf/ __init__.py:1322
LOW	bandit	Starting a process without a shell.	6	/app/ cudatext.app/ Contents/ Resources/py/ cuda_project_manager/ __init__.py:635

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	subprocess call with shell=True seems safe, but may be changed in the future, consider rewriting without shell	4	/app/py/cuda_addonman/work_cudatext_updates__sourceforge.py:19
LOW	bandit	Consider possible security implications associated with the subprocess module.	21	/app/cudatext.app/Contents/Resources/py/cuda_addonman/work_cudatext_updates__sourceforge.py:17
LOW	bandit	Starting a process with a partial executable path	20	/app/cudatext.app/Contents/Resources/py/cuda_palette/cd_plugin_lib.py:286
LOW	bandit	subprocess call - check for execution of untrusted input.	20	/app/cudatext.app/Contents/Resources/py/cuda_addonman/__init__.py:631
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	9	/tools/download_addons.py
LOW	bandit	Use of assert detected. The enclosed code will be removed when compiling to optimised byte code.	63	/app/cudatext.app/Contents/Resources/py/sys/urllib3/util/ssltransport.py:160

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.