

Villain

github.com/t3l3machus/Villain

Scanned 18 September 2026 at 19:37 · Report generated 29 October 2026 · Python · 4,453 stars

OVERALL ASSESSMENT

Medium

Combined score 48 · 35 findings after filtering

Summary

This repository has been flagged as having a medium vulnerability risk of 28.0 and a medium malicious-pattern risk of 20.0, which means the code contains several insecure coding patterns that an attacker could potentially exploit, but it also shows signs that the project may be intentionally designed to be harmful. The most concerning issues are the presence of reverse shell commands in the code, which allow attackers to execute arbitrary commands on a remote server, and the use of weak MD5 hashes for security purposes, both of which represent active threats rather than just poor coding habits. While some findings relate to supply chain risks due to missing version locks for dependencies like Requests and pycryptodomex, these are secondary to the immediate dangers posed by the shell commands and hash weaknesses. The scan did not find evidence that the code is maliciously crafted or specifically designed to cause harm, but rather that it contains dangerous patterns that could be misused if someone with access to this repository were to deploy it.

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 (28)

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

MALICIOUS PATTERN

Medium (20)

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	2	7	10	19
yara	Signatures of malicious behaviour in source files	7	0	3	10
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	5	0	5
semgrep	Insecure coding patterns across many languages	1	0	0	1

5 of 35 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	Semgrep found a bash reverse shell	1	/Core/ payload_templates/ linux/ reverse_tcp/ bash_read_line.py:26
HIGH	yara	YARA rule matched: detect_reverse_shell — Reverse shell command patterns	3	/Core/ payload_templates/ linux/ reverse_tcp/ python3.py
HIGH	bandit	Starting a process with a shell, possible injection detected, security issue.	2	/ Villain.py:161
HIGH	yara	YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed	1	/Core/ villain_core.py
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	/Core/ Utilities/ Shell/ fileless_exec/http/ src.sh

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
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	/Core/villain_core.py
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/Villain.py
MEDIUM	dep_checker	Unpinned dependency: 'netifaces' has no version lock — supply chain risk	1	requirements.txt:1
MEDIUM	dep_checker	Unpinned dependency: 'pyperclip' has no version lock — supply chain risk	1	requirements.txt:3
MEDIUM	dep_checker	Unpinned dependency: 'pycryptodomex' has no version lock — supply chain risk	1	requirements.txt:2
MEDIUM	dep_checker	Unpinned dependency: 'gnureadline' has no version lock — supply chain risk	1	requirements.txt:5
MEDIUM	bandit	Use of weak MD5 hash for security. Consider usedforsecurity=False	2	/Villain.py:490
MEDIUM	dep_checker	Unpinned dependency: 'Requests' has no version lock — supply chain risk	1	requirements.txt:4
MEDIUM	bandit	Possible binding to all interfaces.	5	/Core/settings.py:77
LOW	bandit	Starting a process with a shell: Seems safe, but may be changed in the future, consider rewriting without shell	2	/Villain.py:970
LOW	bandit	Starting a process without a shell.	1	/Villain.py:517
LOW	bandit	Try, Except, Continue detected.	3	/Core/villain_core.py:1814

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Consider possible security implications associated with the subprocess module.	2	/Villain.py:10
LOW	bandit	Starting a process with a partial executable path	2	/Villain.py:970
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	3	/Core/Utilities/Shell/upload/http/src.sh

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 18 September 2026.