

LaZagne

github.com/AlessandroZ/LaZagne

Scanned 21 September 2026 at 17:30 · Report generated 29 October 2026 · Python · 10,988 stars

OVERALL ASSESSMENT

Critical

Combined score 458 · 260 findings after filtering

Summary

This repository has two major security issues that are serious enough to prevent safe use. The most critical problem is that the code relies on outdated and insecure libraries like pyCrypto, which have been deprecated by their developers. Additionally, there are multiple instances of dangerous shell commands being used in subprocess calls, which can lead to unexpected behavior or injection attacks. While some findings suggest the code might be trying to hide malicious intent, the majority of these issues stem from poor security practices rather than intentional attacks. The overall vulnerability risk score is 426.0, indicating a high level of insecurity that requires immediate attention before the code can be considered safe for production use.

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 (426)

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

MALICIOUS PATTERN

Medium (33)

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	15	51	127	193

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
semgrep	Insecure coding patterns across many languages	17	30	0	47
guarddog	Malicious package and install-time behaviour	0	8	5	13
yara	Signatures of malicious behaviour in source files	4	0	1	5
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	2	0	2

134 of 260 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HIGH	bandit	The pyCrypto library and its module AES are no longer actively maintained and have been deprecated. Consider using pyca/cryptography library.	7	/Windows/lazagne/software/ browsers/chromiumbased.py 19
HIGH	bandit	The pyCrypto library and its module unpad are no longer actively maintained and have been deprecated. Consider using pyca/cryptography library.	2	/Windows/lazagne/software/ browsers/chromiumbased.py 21
HIGH	semgrep	The native Python `xml` library is vulnerable to XML External Entity (XXE) attacks. These attacks can leak confidential data and "XML bombs" can cause denial of service. Do not use this library to parse untrusted input. Instead the Python documentation recommends using `defusedxml`.	12	/Windows/lazagne/software/ sysadmin/rdpmanager.py:10
HIGH	semgrep	Found 'subprocess' function 'Popen' with 'shell=True'. This is dangerous because this call will spawn the command using a shell process. Doing so propagates current shell settings and variables, which makes it much easier for a malicious actor to execute commands. Use 'shell=False' instead.	4	/Mac/lazagne/config/ run.py:4

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Found 'subprocess' function 'check_output' with 'shell=True'. This is dangerous because this call will spawn the command using a shell process. Doing so propagates current shell settings and variables, which makes it much easier for a malicious actor to execute commands. Use 'shell=False' instead.	1	/Mac/lazagne/software/browsers/chrome.py:82
HIGH	bandit	Starting a process with a shell, possible injection detected, security issue.	1	/Windows/lazagne/software/sysadmin/iiscentralcert.py:39
HIGH	bandit	subprocess call with shell=True identified, security issue.	5	/Mac/lazagne/config/run.py:4
HIGH	yara	YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed	4	/Windows/lazagne/config/execute_cmd.py
MEDIUM	semgrep	Detected SHA1 hash algorithm which is considered insecure. Use a modern hash algorithm from the SHA-2, SHA-3, or BLAKE2 family instead.	1	/Windows/lazagne/software/browsers/chromiumbased.py:177
MEDIUM	semgrep	It looks like MD5 is used as a password hash. MD5 is not considered a secure password hash because it can be cracked by an attacker in a short amount of time. Use a suitable password hashing function such as scrypt. You can use `hashlib.scrypt`.	2	/Windows/lazagne/software/sysadmin/mRemoteN.py:139
MEDIUM	guarddog	Detects extensive system/network enumeration activities [threat-runtime-enumeration]	3	/Linux/lazagne/config/homes.py:12

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	bandit	Using xml.dom.minidom.parse to parse untrusted XML data is known to be vulnerable to XML attacks. Replace xml.dom.minidom.parse with its defusedxml equivalent function or make sure defusedxml.defuse_stdlib() is called	1	/Windows lazagne/software / sysadmin mRemoteN .py:124
MEDIUM	bandit	Using xml.dom.minidom.parseString to parse untrusted XML data is known to be vulnerable to XML attacks. Replace xml.dom.minidom.parseString with its defusedxml equivalent function or make sure defusedxml.defuse_stdlib() is called	1	/Windows lazagne/software / sysadmin iiscentr alcertp. py:77
MEDIUM	bandit	The pyCrypto library and its module SHA1 are no longer actively maintained and have been deprecated. Consider using pyca/cryptography library.	1	/Windows lazagne/software / browsers chromium based.py 20
MEDIUM	guarddog	Detects access to sensitive files (credentials, configs, keys) [threat-filesystem-read]	5	/Windows lazagne/software /git/ gitforwi ndows.py 49
MEDIUM	bandit	Using xml.etree.ElementTree.parse to parse untrusted XML data is known to be vulnerable to XML attacks. Replace xml.etree.ElementTree.parse with its defusedxml equivalent function or make sure defusedxml.defuse_stdlib() is called	12	/Windows lazagne/software /memory/ libkeepa ss/ common.p :251
MEDIUM	dep_checker	Unpinned dependency: 'pycryptodome' has no version lock — supply chain risk	1	requirem ents.txt 8
MEDIUM	dep_checker	Unpinned dependency: 'pyasn1' has no version lock — supply chain risk	1	requirem ents.txt 3

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected SHA1 hash algorithm which is considered insecure. SHA1 is not collision resistant and is therefore not suitable as a cryptographic signature. Use SHA256 or SHA3 instead.	17	/Linux/lazagne/software/browsers/mozilla.py:313
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>	6	/.github/workflow/WinCompile.yml:1
MEDIUM	bandit	Using <code>xml.etree.ElementTree.fromstring</code> to parse untrusted XML data is known to be vulnerable to XML attacks. Replace <code>xml.etree.ElementTree.fromstring</code> with its <code>defusedxml</code> equivalent function or make sure <code>defusedxml.defuse_stdlib()</code> is called	3	/Windows/lazagne/software/memory/libkeepass/kdb4.py:24
MEDIUM	bandit	Use of weak SHA1 hash for security. Consider <code>usedforsecurity=False</code>	18	/Linux/lazagne/software/browsers/mozilla.py:315
MEDIUM	bandit	Use of weak MD5 hash for security. Consider <code>usedforsecurity=False</code>	12	/Windows/lazagne/software/windows/credump/win32/hashdump.py:196
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.	3	/Windows/lazagne/config/soft_import_module.py:13

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	bandit	Possible SQL injection vector through string-based query construction.	2	/Linux/ lazagne/ config/ lib/ memorpy/ WinProce ss.py:15
MEDIUM	semgrep	Found non static data as an index to 'globals()'. This is extremely dangerous because it allows an attacker to execute arbitrary code on the system. Refactor your code not to use 'globals()'.	1	/Windows lazagne/ software /windows creddump / newobj.p :59
MEDIUM	bandit	Probable insecure usage of temp file/directory.	1	/Linux/ lazagne/ software / browsers chromium based.py 141
LOW	bandit	Using parse to parse untrusted XML data is known to be vulnerable to XML attacks. Replace parse with the equivalent defusedxml package, or make sure defusedxml.defuse_stdlib() is called.	5	/Windows lazagne/ software / sysadmin apachedi rectorys tudio.py 2
LOW	bandit	Using minidom to parse untrusted XML data is known to be vulnerable to XML attacks. Replace minidom with the equivalent defusedxml package, or make sure defusedxml.defuse_stdlib() is called.	1	/Windows lazagne/ software / sysadmin iiscentr alcertp. py:9
LOW	bandit	Possible hardcoded password: 'anonymous'	1	/Windows lazagne/ software / sysadmin ftpnavig ator.py: 0

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Possible hardcoded password: 'qinda'	2	/Windows/lazagne/software/ database/ dbvis.py 20
LOW	bandit	Possible hardcoded password: 'S-1-5-18'	1	/Windows/lazagne/config/ change_privilege.py:112
LOW	bandit	Possible hardcoded password: 'mR3m'	1	/Windows/lazagne/software/ sysadmin/mRemoteN.py:36
LOW	bandit	Possible hardcoded password: 'connection'	2	/Windows/lazagne/software/ multimedia/eyecon.py:61
LOW	bandit	Possible hardcoded password: '{"name":"master-password","value":"(.*)","type":"P","designation":"password"}'	1	/Windows/lazagne/software/memory/onepassword.py:3
LOW	bandit	Possible hardcoded password: 'Password'	8	/Linux/lazagne/software/ database/ dbvis.py 56

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	guarddog	Detects active collection of system information (hostname, platform, architecture, user) [threat-runtime-system-info]	2	/Linux/ lazagne/ config/ write_ou tput.py: 8
LOW	guarddog	Detects memory scraping and credential dumping from process memory [threat-process-memory]	3	/Linux/ lazagne/ config/ lib/ memorpy/ WinProce ss.py:23
LOW	bandit	Possible hardcoded password: '{"name":"account-key","value":'(.{2}-.{6}-.{6}-.{5}-.{5}-.{5}-.{5})',"type":"T"}'	1	/Windows lazagne/ software /memory/ onepassw ord.py:3
LOW	bandit	Using ElementTree to parse untrusted XML data is known to be vulnerable to XML attacks. Replace ElementTree with the equivalent defusedxml package, or make sure defusedxml.defuse_stdlib() is called.	23	/Windows lazagne/ software /wifi/ wifi.py:
LOW	bandit	Possible hardcoded password: '0'	1	/Linux/ lazagne/ software /mails/ clawsmai .py:134
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.	2	/Windows lazagne/ software / sysadmin rdpmanag er.py:10
LOW	bandit	Possible hardcoded password: "	14	/Windows lazagne/ software / sysadmin vnc.py:1 1

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Possible hardcoded password: 'None'	3	/Windows/lazagne/config/DPAPI/masterkey.py:244
LOW	bandit	Try, Except, Continue detected.	16	/Linux/lazagne/config/homes.py:114
LOW	bandit	subprocess call with shell=True seems safe, but may be changed in the future, consider rewriting without shell	2	/Linux/lazagne/config/lib/memorpy/OSXProcess.py:87
LOW	bandit	Consider possible security implications associated with the subprocess module.	10	/Windows/lazagne/software/wifi/wifi.py:
LOW	bandit	Starting a process with a partial executable path	3	/Linux/lazagne/config/lib/memorpy/OSXProcess.py:87
LOW	bandit	subprocess call - check for execution of untrusted input.	4	/Windows/lazagne/software/sysadmin/iisappools.py:27
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	1	/.travis.yml

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	bandit	Use of assert detected. The enclosed code will be removed when compiling to optimised byte code.	26	/Linux/lazagne/software/ browsers/ mozilla. py:327

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 21 September 2026.