

MalwareSourceCode

github.com/vxunderground/MalwareSourceCode

Scanned 29 July 2026 at 15:18 · Report generated 29 October 2026 · Assembly · 18,194 stars

OVERALL ASSESSMENT

Critical

Combined score 498 · 165 findings after filtering

Summary

This repository has been flagged as having a critical vulnerability score of 498.0, which means there are serious insecure coding patterns that an attacker could exploit if the code is used. The scan found many issues related to unsafe file handling and buffer overflows caused by functions like `sprintf` and path traversal vulnerabilities, but these appear to be standard mistakes in how the code was written rather than signs of intentional malicious intent. While some findings involve sensitive AWS credentials or external data that should not be hardcoded, the overall picture suggests the codebase contains significant security gaps that need immediate attention before it can be trusted 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 (498)

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

MALICIOUS PATTERN

Safe (0)

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
semgrep	Insecure coding patterns across many languages	13	112	40	165

164 of 165 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	The application builds a file path from potentially untrusted data, which can lead to a path traversal vulnerability. An attacker can manipulate the path which the application uses to access files. If the application does not validate user input and sanitize file paths, sensitive files such as confi	8	/Linux/Mirai-Family/ Mirai.Linux.eragon.c:1634
HIGH	semgrep	AWS Secret Access Key detected	1	/Python/Trojan-Ransom.Python.Redkeeper.a:20
HIGH	semgrep	The `strchr` function returns NULL on error and this line dereferences the return value without checking for NULL.	3	/Linux/Mirai-Family/ Mirai.Linux.Eragon2.0.c:1785
HIGH	semgrep	AWS Access Key ID Value detected. This is a sensitive credential and should not be hardcoded here. Instead, read this value from an environment variable or keep it in a separate, private file.	1	/Python/Trojan-Ransom.Python.Redkeeper.a:19
MEDIUM	semgrep	Externally controlled data is used to index a fixed-size array, `lin`. This could lead to an out-of-bounds read or write, triggering memory corruption.	2	/Win32/InternetWorm/I-Worm.PieceByPiece.c:504
MEDIUM	semgrep	Avoid using 'strtok()'. This function directly modifies the first argument buffer, permanently erasing the delimiter character. Use 'strtok_r()' instead.	55	/Libs/DDoS/VirTool.DDoS.GHP.c:287
MEDIUM	semgrep	Externally controlled data influences the size of an allocation. This can usually lead to overflow or underflow and later trigger an out of bounds conditions.	29	/Libs/DDoS/VirTool.DDoS.DOMINATE.c:167
MEDIUM	semgrep	The function `sprintf` does not impose any size limitation to what it writes to `request`. That may lead to a stack buffer overflow if there is no validation on the size of the input.	7	/Linux/Mirai-Family/ Mirai.Linux.yakuza.c:1179

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`cmdstr`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	3	/Win32/ Win32.Blasters. cpp:1021
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`parseError`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	3	/Libs/DDoS/ VirTool.DDoS.R UDY.c:1016
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`tempfile`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	1	/MSDOS/V-Index/ Virus.MSDOS.Un known.vlp1.c:31 4
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`CleanDevice`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	1	/Linux/Mirai- Family/ Mirai.Linux.ya kuza.c:938
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`msg_cmd`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	1	/Win32/ InternetWorm/ Worm.Win32.War skype.c:86
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`msg`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	1	/Win32/ InternetWorm/ Worm.Win32.War skype.c:43
MEDIUM	semgrep	The function <code>`sprintf`</code> does not impose any size limitation to what it writes to <code>`myhostname`</code> . That may lead to a stack buffer overflow if there is no validation on the size of the input.	1	/Win32/ Win32.Blasters. cpp:252
MEDIUM	semgrep	Possible active debug code detected. Deploying an application with debug code can create unintended entry points or expose sensitive information.	6	/Java/ Virus.Java.Che shire.a/src/ main/java/ SelfExamine.ja va:476

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected a network listener listening on 0.0.0.0 or an empty string. This could unexpectedly expose the server publicly as it binds to all available interfaces. Instead, specify another IP address that is not 0.0.0.0 nor the empty string.	1	/Linux/Linux.BotenaGo.go:2840
MEDIUM	semgrep	Do not use <code>`math/rand`</code> . Use <code>`crypto/rand`</code> instead.	1	/Linux/Linux.BotenaGo.go:13
LOW	semgrep	Don't call <code>`system`</code> . It's a high-level wrapper that allows for stacking multiple commands. Always prefer a more restrictive API such as calling <code>`execve`</code> from the <code>`exec`</code> family.	38	/Linux/Mirai-Family/Mirai.Linux.Eragon2.0.c:373
LOW	semgrep	This call to <code>`memset`</code> may be optimized out by the compiler because the buffer is not accessed later in the function. This could be an issue if <code>`linebuf`</code> contains sensitive data, such as passwords or cryptographic keys. Applications should use functions such as <code>`memset_s`</code> or <code>`memset_explicit`</code> to ensu	2	/Linux/Mirai-Family/Mirai.Linux.eragon.c:1576

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 July 2026.