# Issue 10421 - mdb_load can crash on malicious input

| | | | |
|---|---|---|---|
| **Status:** | VERIFIED FIXED | **Reported:** | 2026-01-06 20:48 UTC by Howard Chu |
| | | **Modified:** | 2026-01-29 22:01 UTC (History) |
| **Alias:** | None | **CC List:** | 0 users |
| **Product:** | LMDB | | |
| **Component:** | tools (show other issues) | **See Also:** | |
| **Version:** | 0.9.14 | | |
| **Hardware:** | All All | | |
| **Importance:** | --- minor | | |
| **Target Milestone:** | 0.9.34 | | |
| **Assignee:** | OpenLDAP project | | |
| **URL:** | | | |
| **Keywords:** | | | |
| **Depends on:** | | | |
| **Blocks:** | | | |

**Attachments**

Add an attachment (proposed patch, testcase, etc.)

Note

You need to log in before you can comment on or make changes to this issue.

**Howard Chu** **2026-01-06 20:48:08 UTC** Description

This is being reported as a DOS vulnerability, which is incorrect.
https://www.socdefenders.ai/item/01d72f7a-9622-4384-a936-d99925a19a8f

mdb_load is a command-line tool, not a server nor a library function that could be run in a server. There is no service to deny.

Reading a line containing an embedded NUL byte may cause mdb_load to crash, but as it's a one-shot commandline tool such crashes have no consequences.

The report claims "heap metadata leak" which is also irrelevant since this is a one-shot tool and the only potential metadata is that which was contained in the input file, whose contents are already known to the attacker.

The report is supposedly released under "responsible disclosure" and yet it was never reported directly to the OpenLDAP Project (not in this bug tracker nor anywhere else). This despite the fact that the reporter clearly knows that mdb_load is a piece of OpenLDAP software. https://seclists.org/fulldisclosure/2026/Jan/5

The malicious input file must be constructed with a valid header and a line containing a single space followed by a single NUL byte. The dump/load file format only uses printable characters, so an embedded NUL byte never occurs in valid files.

It is unclear why any DB admin would ever fall victim to such a malicious input. The mdb_load utility is only used to load files generated by mdb_dump, and mdb_dump will never produce such an invalid file. If an attacker went to the trouble to

binary patch a dump file to create this crashing character sequence, they clearly have enough privileges to read and alter any other relevant files, so this crash doesn't give them any particular advantage. Nor does the crash reveal any memory content that they don't already know.

**Howard Chu** **2026-01-06 20:57:00 UTC** Comment 1

Fixed in git 8e1fda85532a3c74276df38a42d234dcdfa1e40d etc

**Quanah Gibson-Mount** **2026-01-13 00:05:36 UTC** Comment 2

mdb.master:
  • 8e1fda85
by Howard Chu at 2026-01-06T20:52:25+00:00
ITS#10421 mdb_load: check for malicious input

mdb.master3:
 • 7b136c42
by Howard Chu at 2026-01-06T20:53:01+00:00
ITS#10421 mdb_load: check for malicious input

mdb.RE/0.9:
  • 46b33d2b
by Howard Chu at 2026-01-06T20:53:08+00:00
ITS#10421 mdb_load: check for malicious input

**Howard Chu** **2026-01-15 02:13:07 UTC** Comment 3

A CVE has been issued for this bug https://www.cve.org/CVERecord?id=CVE-2026-22185

We have disputed the validity of this report. The bug does not cause a denial of service condition. The bug does not reliably cause mdb_load to crash. Even on an unpatched mdb_load, the invalid input is flagged and the mdb_load utility exits cleanly with an error message, there is no crash on any of the supported platforms using various C compilers, optimization flags, and the standard libc malloc library. It is only when using a diagnostic library (e.g. Efence, or compiling with ASAN) that a fault or assert is triggered.

The behavior both with and without the fix is the same: the mdb_load utility prints an error message and exits.

The reporter argues that even though mdb_load is a one-shot commandline utility, it can be considered a denial of service if the tool is used in automated scripts. But that's a bogus argument, since the utility would exit with an error anyway, when fed invalid input. The actual security exposure is in whatever system processes allowed an invalid input file to be created and fed to the tool.

Format For Printing - XML - Clone This Issue - Top of page