Buffer Over-read in bfabiszewski/libmobi

0

Valid Reported on Apr 25th 2022

Description

Stack-based Buffer Overflow at index.c:991

Build

```
git clone https://github.com/bfabiszewski/libmobi.git
cd libmobi

export CFLAGS="-g -O0 -lpthread -fsanitize=address"
export CXXFLAGS="-g -O0 -lpthread -fsanitize=address"
export LDFLAGS="-fsanitize=address"

./autogen.sh

./configure --disable-shared

make
```

POC

```
./tools/mobitool -e -o ./tmp/ ./poc_s.mobi
```

poc_s.mobi

Asan

```
Title: Libmobi sample file
Author: Bartek Fabiszewski
```

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

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

Reconstructing source resources...
=================================================================
==1384948==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7fff
READ of size 1 at 0x7fffffffb9ff thread T0
    #0 0x59774c in mobi_decode_infl /home/fuzz/libmobi/src/index.c:991:21
    #1 0x4f8de3 in mobi_reconstruct_infl /home/fuzz/libmobi/src/parse_rawml
    #2 0x4fabbc in mobi_reconstruct_orth /home/fuzz/libmobi/src/parse_rawml
    #3 0x4fd1fb in mobi_reconstruct_links_kf7 /home/fuzz/libmobi/src/parse_
    #4 0x4fd916 in mobi_reconstruct_links /home/fuzz/libmobi/src/parse_rawm
    #5 0x5011d3 in mobi_parse_rawml_opt /home/fuzz/libmobi/src/parse_rawml.
    #6 0x4ff78f in mobi_parse_rawml /home/fuzz/libmobi/src/parse_rawml.c:26
    #7 0x4c98d4 in loadfilename /home/fuzz/libmobi/tools/mobitool.c:852:20
    #8 0x4c8b36 in main /home/fuzz/libmobi/tools/mobitool.c:1051:11
    #9 0x7ffff7a7a0b2 in __libc_start_main /build/glibc-sMfBJT/glibc-2.31/c
    #10 0x41d57d in _start (/home/fuzz/libmobi/tools/mobitool+0x41d57d)

Address 0x7fffffffb9ff is located in stack of thread T0 at offset 1279 in f
    #0 0x4f7fef in mobi_reconstruct_infl /home/fuzz/libmobi/src/parse_rawml

  This frame has 7 object(s):
    [32, 40) 'infl_groups' (line 1366)
    [64, 565) 'name_attr' (line 1375)
    [640, 1141) 'infl_tag' (line 1376)
    [1216, 1224) 'groups' (line 1395)
    [1248, 1256) 'parts' (line 1397)
    [1280, 1781) 'decoded' (line 1414) <== Memory access at offset 1279 unc
    [1856, 1860) 'decoded_length' (line 1418)
HINT: this may be a false positive if your program uses some custom stack u
      (longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-overflow /home/fuzz/libmobi/src/inc
Shadow bytes around the buggy address:
  0x10007fff76e0: 00 00 00 00 00 00 05 f2 f2 f2 f2 f2 f2 f2
  0x10007fff76f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
  0x10007fff7700: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007fff7710: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007fff7720: 00 00 00 00 00 00 00 00 00 00 00 00 00 05 f2

=>0x10007fff7730: f2 f2 f2 f2 f2 f2 f2 f2 00 f2 f2 f2 00 f2 f2[f2]
  0x10007fff7740: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007fff7750: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007fff7760: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007fff7770: 00 00 00 00 00 00 00 00 00 00 00 00 00 05 f2
  0x10007fff7780: f2 f2 f2 f2 f2 f2 f2 f2 04 f3 f3 f3 00 00 00 00
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:              cc
==1384948==ABORTING
```

Impact

This vulnerability is capable of arbitrary code execution.

CVE
CVE-2022-1533
(Published)

Vulnerability Type
CWE-126: Buffer Over-read

Severity
Medium (6.6)

Registry
Other

Affected Version
*

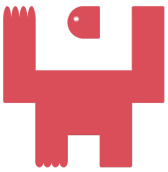
Visibility
Public

Status
Fixed

Found by

TDHX ICS Security
@jieyongma
pro

Fixed by



Bartek Fabiszewski
@bfabiszewski
unranked

This report was seen 632 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 validated this vulnerability 7 months ago

TDHX ICS Security has been awarded the disclosure bounty ✔

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

Bartek Fabiszewski marked this as fixed in **0.11** with commit **eafc41** 7 months ago

Bartek Fabiszewski has been awarded the fix bounty ✔

This vulnerability will not receive a CVE ✖

Bartek 7 months ago — Maintainer

Thanks!

Jamie Slome 7 months ago — Admin

@maintainer - the researcher has requested a CVE for this report and another report. Are you happy for us to proceed with assigning and publishing a CVE for these two reports?

Bartek 7 months ago — Maintainer

I just wonder, should both these issues be classified as buffer overflows? Technically these are rather buffer over-reads if that matters. But I am not familiar with CVE's vulnerability categories.

Anyway feel free to proceed with CVE.

Jamie Slome 7 months ago — Admin

@maintainer - I'm happy to adjust the CWE (vulnerability type) to Buffer Over-read (CWE-126) if you think both this issue and the other fall under this category.

Let me know if you would like me to proceed with this 👍

Bartek 7 months ago — Maintainer

@jamie Yes, I think that is more relevant. Thanks!

Jamie Slome 7 months ago

Sorted for both 👍

Jamie Slome 7 months ago Admin

CVEs also arranged for both reports!

Bartek 7 months ago Maintainer

Thanks!

ajakk 7 months ago

How could this result in arbitrary code execution?

> Sign in to join this conversation