New issue

Jump to bottom

global-buffer-overflow in function jfif_encode at jfif.c:701 #25

Closed **WayneDevMaze** opened this issue on Jun 22, 2020 · 1 comment

WayneDevMaze commented on Jun 22, 2020

Describe

A global-buffer-overflow was discovered in ffjpeg. The issue is being triggered in function jfif_encode at jfif.c:701

Reproduce

Tested in Ubuntu 18.04, 64bit. Compile ffjpeg with address sanitizer as I changed `CCFLAGS` src/Makefile as:

```
# ASan
CCFLAGS = -Wall -g -fsanitize=address -fno-omit-frame-pointer -O1
```

And do this command to reproduce this issue:
`ffjpeg -e $poc`
poc is here

Expected behavior

An attacker can exploit this vulnerability by submitting a malicious jpeg that exploits this issue. This will result in a Denial of Service (DoS) and when the application attempts to process the file.

ASAN Reports

```
==81140==ERROR: AddressSanitizer: global-buffer-overflow on address 0x56063a6c1b92 at pc 0x7f2c96918733 bp 0x7ffe503e7fe0 sp 0x7ffe503e7788
READ of size 272 at 0x56063a6c1b92 thread T0
    #0 0x7f2c96918732  (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x79732)
    #1 0x56063a6a88e7 in memcpy /usr/include/x86_64-linux-gnu/bits/string_fortified.h:34
    #2 0x56063a6a88e7 in jfif_encode /root/study/ffjpeg/src/jfif.c:701
    #3 0x56063a69b382 in main /root/study/ffjpeg/src/ffjpeg.c:30
    #4 0x7f2c964cfb96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21b96)
    #5 0x56063a69b829 in _start (/root/study/ffjpeg/test/ffjpeg+0x2829)

0x56063a6c1b92 is located 0 bytes to the right of global variable 'STD_HUFTAB_LUMIN_AC' defined in 'huffman.c:388:12' (0x56063a6c1ae0) of size 178
SUMMARY: AddressSanitizer: global-buffer-overflow (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x79732)
Shadow bytes around the buggy address:
  0x0ac1474d0320: 00 00 00 00 00 00 00 00 05 f9 f9 f9 f9 f9 f9 f9
  0x0ac1474d0330: 00 00 00 04 f9 f9 f9 f9 00 00 00 00 00 00 00 00
  0x0ac1474d0340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 f9
  0x0ac1474d0350: f9 f9 f9 f9 00 00 00 04 f9 f9 f9 f9 00 00 00 00
  0x0ac1474d0360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0ac1474d0370: 00 00[02]f9 f9 f9 f9 f9 00 00 00 00 00 00 00 00
  0x0ac1474d0380: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0ac1474d0390: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0ac1474d03a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0ac1474d03b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0ac1474d03c0: 00 00 00 00 00 00 00 00 00 00 00 00 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
==81140==ABORTING
```

rockcarry commented on Jul 27, 2020 Owner

lastest code can't reproduce this issue.
last commit is: `31649ad`
@**WayneDevMaze** please check and test.

🎉 1

WayneDevMaze closed this as completed on Aug 6, 2020

Marsman1996 mentioned this issue on Nov 30, 2021

global-buffer-overflow in function jfif_encode at jfif.c:708 #44

Closed

rockcarry added a commit that referenced this issue on Dec 1, 2021

`Merge pull request #45 from Marsman1996/master` … `23dffdf`

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants