

New issue

Jump to bottom

report a stack buffer overflow security issue #392

Closed

lxkektx opened this issue on Dec 5, 2019 · 1 comment

Assignees



Labels

bug fixed

lxkektx commented on Dec 5, 2019

There's a stack buffer overflow in encode_one_block function, the backtrace of the crash point is below. The version is the latest from the git main branch:

```
./jpegtran-static -transverse ~/Desktop/poc.jpg

=====
==11652==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffd03285910 at pc 0x00000051103c bp 0x7ffd03285700 sp 0x7ffd032856f0
WRITE of size 1 at 0x7ffd03285910 thread T0
#0 0x51103b in encode_one_block /home/zwjj/research/libjpeg-turbo/jchuff.c:600
#1 0x51103b in encode_mcu_huff /home/zwjj/research/libjpeg-turbo/jchuff.c:684
#2 0x475668 in compress_output /home/zwjj/research/libjpeg-turbo/jctrans.c:349
#3 0x4443da in jpeg_finish_compress /home/zwjj/research/libjpeg-turbo/jcapimin.c:188
#4 0x404e14 in main /home/zwjj/research/libjpeg-turbo/jpegtran.c:581
#5 0x7fbfb2ed182f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)
#6 0x405a58 in _start (/home/zwjj/research/libjpeg-turbo/jpegtran-static+0x405a58)

Address 0x7ffd03285910 is located in stack of thread T0 at offset 384 in frame
#0 0x4d839f in encode_mcu_huff /home/zwjj/research/libjpeg-turbo/jchuff.c:648

This frame has 2 object(s):
[32, 88) 'state'
[128, 384) '_buffer' <== Memory access at offset 384 overflows this variable
HINT: this may be a false positive if your program uses some custom stack unwind mechanism or swapcontext
(longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-overflow /home/zwjj/research/libjpeg-turbo/jchuff.c:600 encode_one_block
Shadow bytes around the buggy address:
 0x100020648ad0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x100020648ae0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x100020648af0: 00 00 f1 f1 f1 00 00 00 00 00 00 f4 f2 f2
 0x100020648b00: f2 f2 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x100020648b10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x100020648b20: 00 00[f3]f3 f3 f3 f3 f3 f3 00 00 00 00 00 00
 0x100020648b30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x100020648b40: 00 00 00 00 00 00 00 00 00 00 f1 f1 f1 00 00
 0x100020648b50: 00 00 f2 f2 f2 00 00 00 00 00 00 00 00 00
 0x100020648b60: f4 f4 f3 f3 f3 00 00 00 00 00 00 00 00 00
 0x100020648b70: 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
Heap right redzone: fb
Freed heap region: fd
Stack left redzone: f1
Stack mid redzone: f2
Stack right redzone: f3
Stack partial redzone: f4
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
==11652==ABORTING
```

poc

lxkektx added the bug label on Dec 5, 2019

lxkektx assigned dcommander on Dec 5, 2019


dcommander closed this as completed in c76f4a0 on Dec 5, 2019

dcommander added the fixed label on Dec 5, 2019

dcommander commented on Dec 5, 2019

Member

I don't believe that this issue was exploitable, since the overrun was fully contained in the stack (valgrind didn't even detect it) and since the lossless transformer is, unlike the decompressor, not generally exposed to arbitrary data exploits (i.e. it is generally used offline, not online.) Fixed and pushed.

 **dcommander** added a commit that referenced this issue on Jun 3, 2020

 Huffman enc.: Fix very rare local buffer overrun ...

6bbc0a3

  **dcommander** mentioned this issue on Jun 18, 2021

Libjpeg-turbo all version have a stack-based buffer overflow in the "transform" component. A remote attacker can send a malformed jpeg file to the service and cause arbitrary code execution or denial of service of the target service. #527

 Closed

Assignees

 **dcommander**

Labels

bug fixed

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants

