LibreDWG "read_system_page" function heap overflow vulnerability #248

Closed yangjiageng opened this issue on Jul 18, 2020 · 6 comments

Assignees	
Labels	bug fuzzing
Milestone	0.11

yangjiageng commented on Jul 18, 2020 • edited

LibreDWG "read_system_page" function heap overflow vulnerability

Description:
There is a heap overflow bug in "read_system_page" function at libredwg-0.10.1/src/decode_r2007.c:666:5
An attacker can exploit this bug to cause a Denial of Service (DoS) by submitting a dwg file.
This bug is caused by the following dangerous memcpy calling in read_system_page function: line 666

```
if (size_comp < size_uncomp)
    error = decompress_r2007 (data, size_uncomp, pedata, MIN (pedata_size, size_comp));
  else
    memcpy (data, pedata, size_uncomp);
  free (pedata);
```

We used AddressSanitizer instrumented in LibreDWG and triggered this bug, the asan output as follows:

```
==2593==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000000031 at pc 0x0000004e7e12 bp 0x7ffd91a390c0 sp 0x7ffd91a38870
READ of size 96 at 0x602000000031 thread T0
#0 0x4e7e11 in __asan_memcpy /root/Download/llvm-8.0.0.src/projects/compiler-rt-8.0.0.src/lib/asan/asan_interceptors_memintrinsics.cc:23
#1 0x7f131338293e in read_system_page /root/libredwg-0.10.1/src/decode_r2007.c:666:5
#2 0x7f131336bc15 in read_pages_map /root/libredwg-0.10.1/src/decode_r2007.c:1014:10
#3 0x7f131336bc15 in read_r2007_meta_data /root/libredwg-0.10.1/src/decode_r2007.c:1814
#4 0x7f1312f9bc84 in decode_R2007 /root/libredwg-0.10.1/src/decode.c:3016:11
#5 0x7f1312f9bc84 in dwg_decode /root/libredwg-0.10.1/src/decode.c:241
#6 0x7f1312f7922f in dwg_read_file /root/libredwg-0.10.1/src/dwg.c:211:11
#7 0x52c287 in main /root/libredwg-0.10.1/programs/dwg2dxf.c:255:15
#8 0x7f1311da1b96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21b96)
#9 0x41add9 in _start (/root/libredwg-0.10.1/programs/.libs/dwg2dxf+0x41add9)

0x602000000031 is located 0 bytes to the right of 1-byte region [0x602000000030,0x602000000031)
allocated by thread T0 here:
#0 0x4e93cf in calloc /root/Download/llvm-8.0.0.src/projects/compiler-rt-8.0.0.src/lib/asan/asan_malloc_linux.cc:155
#1 0x7f131337a0a6 in decode_rs /root/libredwg-0.10.1/src/decode_r2007.c:590:34

SUMMARY: AddressSanitizer: heap-buffer-overflow /root/Download/llvm-8.0.0.src/projects/compiler-rt-8.0.0.src/lib/asan/asan_interceptors_memintrinsics.cc:23 in __asan_memcpy
Shadow bytes around the buggy address:
0x0c047fff7fb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7fc0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7fd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7fe0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7ff0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c047fff8000: fa fa 00 00 fa fa[01]fa fa fa fa fa fa fa fa fa
0x0c047fff8010: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c047fff8020: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c047fff8030: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c047fff8040: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c047fff8050: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
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
==2593==ABORTING
```

Then, we used GDB to debug this bug, the GDB outputs:

```
GDB
//---------------------------------registers----------------------------------
RAX: 0x7ffff2c6a1e0 --> 0x7ffff6fed930 (<read_system_page+976>: dec DWORD PTR [rcx+rcx*4-0x1])
RBX: 0x7f
RCX: 0x0
RDX: 0x60 ('`')
RSI: 0x602000000070 --> 0x0
RDI: 0x608000000020 --> 0x0
RBP: 0x0
RSP: 0x7fffffffc410 --> 0xffffffff866 --> 0x0
RIP: 0x7ffff6fed93a (<read_system_page+986>: call 0x7ffff6be3460 __asan_memcpy@plt)
R8 : 0x1c2e6
R9 : 0x2
R10: 0x28 ('(')
R11: 0x602000000060 --> 0x2ffffff00000002
R12: 0x602000000070 --> 0x0
R13: 0x608000000080 --> 0x0
R14: 0x60 ('`')
R15: 0x608000000020 --> 0x0
EFLAGS: 0x202 (carry parity adjust zero sign trap INTERRUPT direction overflow)
//-----------------------------------code------------------------------------
0x7ffff6fed931 <read_system_page+977>: mov rdi,r15
0x7ffff6fed934 <read_system_page+980>: mov rsi,r12
0x7ffff6fed937 <read_system_page+983>: mov rdx,r14
=> 0x7ffff6fed93a <read_system_page+986>: call 0x7ffff6be3460 __asan_memcpy@plt
0x7ffff6fed93f <read_system_page+991>: mov rdi,r12
0x7ffff6fed942 <read_system_page+994>: call 0x7ffff6be25e0 free@plt
0x7ffff6fed947 <read_system_page+999>: jmp 0x7ffff6fed99a <read_system_page+1082>
0x7ffff6fed949 <read_system_page+1001>: lea rdi,[rip+0xd9a81c] # 0x7ffff7d8816c
Guessed arguments:
arg[0]: 0x608000000020 --> 0x0
arg[1]: 0x602000000070 --> 0x0
arg[2]: 0x60 ('`')
//----------------------------------stack-----------------------------------
0000| 0x7fffffffc410 --> 0xffffffff866 --> 0x0
0008| 0x7fffffffc418 --> 0x7fffffffd098 --> 0x2f65f
0016| 0x7fffffffc420 --> 0x7fffffffd090 --> 0x7ffff7e18800 --> 0x313230314341 ('AC1021')
0024| 0x7fffffffc428 --> 0xffffffffa13 --> 0x0
0032| 0x7fffffffc430 --> 0x7f
0040| 0x7fffffffc438 --> 0x60 ('`')
0048| 0x7fffffffc440 --> 0xffffffffffffff91
0056| 0x7fffffffc448 --> 0x7fffffffcc80 --> 0xf
//--------------------------------------------------------------------------
Legend: code, data, rodata, value
0x00007ffff6fed93a 666 memcpy (data, pedata, size_uncomp);
gdb-peda$

ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000000071 at pc 0x0000004e7e12 bp 0x7fffffffc400 sp 0x7fffffffbbb0
READ of size 96 at 0x602000000071 thread T0
[New process 3499]
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
process 3499 is executing new program: /opt/llvm/bin/llvm-symbolizer
Error in re-setting breakpoint 2: No symbol table is loaded. Use the "file" command.
Warning:
Cannot insert breakpoint 1.
Cannot access memory at address 0x52c283
```

We ensured there is a heap overflow vulnerability because of the dangerous using of memcpy function, attacker can use this bug to finish a DoS attack.

You can reproduce this heap overflow vulnerability by the follow step:
/dwg2dxf -m -b PoC_libreDWG_heapoverflow_decode_r2007_line666

rurban commented on Jul 18, 2020 Contributor

I've fixed hundreds of such bugs in master. There's a 2nd repo called libredwg-fuzz on GitHub with all the test cases. I'm not really interested in 0.10.1 cases as they are all fixed long time ago, and current fuzzing is going on for weeks without any crash.

rurban mentioned this issue on Jul 18, 2020

LibreDWG "read_2004_compressed_section" function heap overflow vulnerability #249
Closed

rurban commented on Jul 18, 2020 • edited Contributor

Please attach the file also, otherwise I cannot verify that its fixed.

rurban added the **fuzzing** label on Jul 18, 2020

yangjiageng commented on Jul 19, 2020 Author

This bug also exists on the master version, the poc is here.
PoC

rurban commented on Jul 19, 2020 • edited Contributor

Thanks, excellent! Good catch.
With which fuzzer was this found? libfuzzer, afl, afl++?

rurban added a commit that referenced this issue on Jul 19, 2020

`decode: protect r2007 empty system_page map` ··· ✓ `a8a360a`

rurban self-assigned this on Jul 19, 2020

rurban added the bug label on Jul 19, 2020

rurban added this to the **0.11** milestone on Jul 19, 2020

rurban closed this as completed on Jul 19, 2020

---

yangjiageng commented on Jul 20, 2020 Author

I develop and use a new fuzzer named PathTracer. I will open source this fuzzer after its paper is accepted.

rurban commented on Jul 21, 2020 Contributor

Oh nice. Seems to be good one. My fuzzers didn't find it.

Jager Yeung <notifications@github.com> schrieb am Di., 21. Juli 2020, 02:45:
…

Assignees
rurban

Labels
bug fuzzing

Projects
None yet

Milestone
0.11

Development
No branches or pull requests

2 participants