qpdf / qpdf Public

Code Issues 121 Pull requests 5 Discussions Actions Projects •••

New issue Jump to bottom

heap-buffer-overflow in `QPDF::processXRefStream` found by ASAN #701

Closed chinggg opened this issue on May 11 · 6 comments

chinggg commented on May 11 • edited

Hi, I have found a heap-buffer-overflow in QPDF 8.4.2 using ASAN, which has not been reported yet. But after `d71f05c` , the heap-buffer-overflow seems to disappear. I don't know exactly how the commit mitigate the problem since it was intended to fix sign and conversion warnings.

To reproduce, compile QPDF with address sanitizer

```
export CC=clang
export CXX=clang++
export CFLAGS="-g -fsanitize=address"
export CPPFLAGS="-g -fsanitize=address"
export LDFLAGS="-fsanitize=address"
```

Download the testcase HeapBOF-processXRefStream.zip

```
qpdf ./HeapBOF-processXRefStream -
```

Then after some warnings, ASAN should complain about heap-buffer-overflow.

```
❯ ./install/bin/qpdf ./HeapBOF-processXRefStream -
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): unknown token while
reading object; treating as string
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 189): invalid character (i)
in hexstring
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 201): unknown token while
reading object; treating as string
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 235): unexpected >
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 269): unknown token while
reading object; treating as string
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 273): unknown token while
reading object; treating as string
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 302): unknown token while
reading object; treating as string
```

```
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 320): unexpected )
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 380): treating unexpected array close token as null
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake1
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake2
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake3
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake4
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake5
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake6
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake7
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected dictionary key but found non-name object; inserting key /QPDFFake8
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 124): /Length key in stream dictionary is not an integer
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 391): attempting to recover stream length
WARNING: ./HeapBOF-processXRefStream (xref stream: object 24 0, offset 391): recovered stream length: 31
WARNING: ./HeapBOF-processXRefStream (xref stream, offset 116): Cross-reference stream data has the wrong size; expected = 35; actual = 38
=================================================================
==2770728==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x604000008936 at pc 0x561e09240f1c bp 0x7ffc29847050 sp 0x7ffc29847048
READ of size 1 at 0x604000008936 thread T0
    #0 0x561e09240f1b in QPDF::processXRefStream(long long, QPDFObjectHandle&) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:1132:33
    #1 0x561e092362f4 in QPDF::read_xrefStream(long long) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:970:20
    #2 0x561e0922415a in QPDF::read_xref(long long) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:598:20
    #3 0x561e09220faf in QPDF::parse(char const*) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:403:2
    #4 0x561e0921f54a in QPDF::processFile(char const*, char const*) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:204:5
    #5 0x561e091d759f in PointerHolder<QPDF> do_process_once<char const*>(void (QPDF::*)(char const*, char const*), char const*, char const*, Options&, bool) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:4005:9
    #6 0x561e091c69cd in PointerHolder<QPDF> do_process<char const*>(void (QPDF::*)(char const*, char const*), char const*, char const*, Options&, bool) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:4043:16
    #7 0x561e091c69cd in process_file(char const*, char const*, Options&) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:4105:12
    #8 0x561e091bd6de in realmain(int, char**) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:5062:13
    #9 0x561e091d73f8 in main qpdf/qpdf-8.4.2/qpdf/qpdf.cc:5143:12
    #10 0x7f9220def30f in __libc_start_call_main libc-start.c
    #11 0x7f9220def3c0 in __libc_start_main@GLIBC_2.2.5 (/usr/lib/libc.so.6+0x2d3c0)
    #12 0x561e0908ce84 in _start (qpdf/qpdf-8.4.2/install/bin/qpdf+0x72e84)

0x604000008936 is located 0 bytes to the right of 38-byte region [0x604000008910,0x604000008936)
allocated by thread T0 here:
    #0 0x561e09171491 in operator new[](unsigned long) (qpdf/qpdf-8.4.2/install/bin/qpdf+0x157491)
    #1 0x561e09415c90 in Buffer::init(unsigned long, unsigned char*, bool) qpdf/qpdf-8.4.2/libqpdf/Buffer.cc:45:22
    #2 0x561e09415c90 in Buffer::Buffer(unsigned long) qpdf/qpdf-8.4.2/libqpdf/Buffer.cc:12:5
    #3 0x561e094201bf in Pl_Buffer::getBuffer() qpdf/qpdf-8.4.2/libqpdf/Pl_Buffer.cc:72:21
    #4 0x561e09378e16 in QPDF_Stream::getStreamData(qpdf_stream_decode_level_e) qpdf/qpdf-
8.4.2/libqpdf/QPDF_Stream.cc:171:16
    #5 0x561e092b4495 in QPDFObjectHandle::getStreamData(qpdf_stream_decode_level_e) qpdf/qpdf-
8.4.2/libqpdf/QPDFObjectHandle.cc:1041:31
    #6 0x561e0923e592 in QPDF::processXRefStream(long long, QPDFObjectHandle&) qpdf/qpdf-
8.4.2/libqpdf/QPDF.cc:1086:41
    #7 0x561e092362f4 in QPDF::read_xrefStream(long long) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:970:20
    #8 0x561e0922415a in QPDF::read_xref(long long) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:598:20
    #9 0x561e09220faf in QPDF::parse(char const*) qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:403:2
    #10 0x561e0921f54a in QPDF::processFile(char const*, char const*) qpdf/qpdf-
8.4.2/libqpdf/QPDF.cc:204:5
    #11 0x561e091d759f in PointerHolder<QPDF> do_process_once<char const*>(void (QPDF::*)(char
const*, char const*), char const*, char const*, Options&, bool) qpdf/qpdf-
8.4.2/qpdf/qpdf.cc:4005:9
    #12 0x561e091c69cd in PointerHolder<QPDF> do_process<char const*>(void (QPDF::*)(char const*,
char const*), char const*, char const*, Options&, bool) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:4043:16
    #13 0x561e091c69cd in process_file(char const*, char const*, Options&) qpdf/qpdf-
8.4.2/qpdf/qpdf.cc:4105:12
    #14 0x561e091bd6de in realmain(int, char**) qpdf/qpdf-8.4.2/qpdf/qpdf.cc:5062:13
    #15 0x561e091d73f8 in main qpdf/qpdf-8.4.2/qpdf/qpdf.cc:5143:12
    #16 0x7f9220def30f in __libc_start_call_main libc-start.c

SUMMARY: AddressSanitizer: heap-buffer-overflow qpdf/qpdf-8.4.2/libqpdf/QPDF.cc:1132:33 in
QPDF::processXRefStream(long long, QPDFObjectHandle&)
Shadow bytes around the buggy address:
  0x0c087fff90d0: fa fa fd fd fd fd fd fa fa fa fd fd fd fd fd fa
  0x0c087fff90e0: fa fa fd fd fd fd fd fa fa fa fd fd fd fd fd fa
  0x0c087fff90f0: fa fa fd fd fd fd fd fa fa fa fd fd fd fd fd fa
  0x0c087fff9100: fa fa fd fd fd fd fd fa fa fa fd fd fd fd fd fa
  0x0c087fff9110: fa fa fd fd fd fd fd fa fa fa fd fd fd fd fd fa
=>0x0c087fff9120: fa fa 00 00 00 00[06]fa fa fa fa fa fa fa fa fa
  0x0c087fff9130: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c087fff9140: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c087fff9150: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c087fff9160: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c087fff9170: 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
==2770728==ABORTING
```

I find https://github.com/qpdf/qpdf/blob/release-qpdf-8.4.2/libqpdf/QPDF.cc#L1111-L1134 responsible for this heap-buffer-overflow.

After reading and debugging this code, I try to explain the reason as below.

`data` points to a piece of previously allocated buf with size of 38, which equals to `actual_size`. `expected_size = entry_size * num_entries` = 5 * 7 = 35, and the expected_size < actual_size is just warn and will not throw an exception.

In each time of the first loop, `entry += entry_size`, then `p = entry`. The second loop iterates j in range(0,3), the innermost loop and then increase W[j] to p, the final overflow exceeds the size of 38, triggering heap-buffer-overflow.

jberkenbilt commented on May 12 Contributor

8.4.2 is very old. Does this happen with 10.6.3 or, better yet, with the tip of main?

zdohnal commented on Jul 27 Contributor

I cannot reproduce the issue with the latest qpdf main branch, nor with 10.6.3 or 10.3.2 (qpdf versions in stable Fedoras).

My compilation/testing steps:

Versions 10.6.3 and 10.3.2:

```
$ export CC=clang; export CXX=clang++; export CFLAGS="-g -fsanitize=address"; export CPPFLAGS="-g -fsanitize=address"; export LDFLAGS="-fsanitize=address"; ./configure --disable-static --enable-crypto-gnutls --disable-implicit-crypto --enable-show-failed-test-output && make && ./qpdf/build/qpdf ~/HeapBOF-processXRefStream -
```

Current main branch:

```
$ export CC=clang; export CXX=clang++; export CFLAGS="-g -fsanitize=address"; export CPPFLAGS="-g -fsanitize=address"; export LDFLAGS="-fsanitize=address"; cmake -S . -B build -DBUILD_STATIC_LIBS=0 -DREQUIRE_CRYPTO_NATIVE=0 -DREQUIRE_CRYPTO_GNUTLS=1 && cmake --build build -j$(nproc) -- -k && ./build/qpdf/qpdf ~/HeapBOF-processXRefStream -
```

zdohnal commented on Jul 27 Contributor

Output for the current main:

```
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): unknown token
while reading object; treating as string
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 189): invalid character
(i) in hexstring
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 201): unknown token
while reading object; treating as string
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 235): unexpected >
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 269): unknown token
while reading object; treating as string
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 273): unknown token
while reading object; treating as string
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 302): unknown token
while reading object; treating as string
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 320): unexpected )
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 380): treating
unexpected array close token as null
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake1
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake2
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake3
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake4
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake5
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake6
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake7
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 127): expected
dictionary key but found non-name object; inserting key /QPDFFake8
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 124): /Length key in
stream dictionary is not an integer
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 391): attempting to
recover stream length
WARNING: /root/HeapBOF-processXRefStream (xref stream: object 24 0, offset 391): recovered stream
length: 31
WARNING: /root/HeapBOF-processXRefStream: file is damaged
WARNING: /root/HeapBOF-processXRefStream: error reading xref: integer out of range converting -4
from a 4-byte signed type to a 8-byte unsigned type
WARNING: /root/HeapBOF-processXRefStream: Attempting to reconstruct cross-reference table
qpdf: /root/HeapBOF-processXRefStream: unable to find trailer dictionary while recovering damaged
file
```

pgajdos commented on Jul 27

`W[1]` happens to be negative (`-4`) for this case, which is guarded with `toS()` in current versions, unlike the older ones.
See https://bugzilla.suse.com/show_bug.cgi?id=1201830#c5 for details and welcome to advice.

👍 1

chinggg commented on Jul 27 Author

Thank you all for the investigation and analysis. Feel free to close the issue if you think the vulnerability has been fixed.

jberkenbilt commented on Jul 27 Contributor

Thanks, all, for helping out. I'll go ahead and close.

jberkenbilt closed this as completed on Jul 27

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

4 participants