DWA Lossy Decoder Heap Out-of-Bounds Write

High **cary-ilm** published **GHSA-p8xc-w3q4-h64x** yesterday

Package

No package listed

Affected versions

3.2.0-3.2.6, 3.3.0-3.3.8, 3.4.0-3.4.8

Patched versions

3.2.7, 3.3.9, 3.4.9

Description

Summary

The DWA lossy decoder constructs temporary per-component block pointers using signed 32-bit arithmetic. For a large enough width, the calculation overflows and later decoder stores operate on a wrapped pointer outside the allocated `rowBlock` backing store.

This bug is reachable from the public decoder path and can be reproduced through the shipped `exrcheck` tool with a crafted scanline DWAA file. The confirmed dynamic symptom is a write-side crash in the lossy DCT execution path.

Tested on commit: `7820b7e`

Root Cause and Data Flow

The vulnerable pointer construction lives in `src/lib/OpenEXRCore/internal_dwa_decoder.h`:

```
for (int comp = 1; comp < numComp; ++comp)
    rowBlock[comp] = rowBlock[comp - 1] + numBlocksX * 64;
```

The expression `numBlocksX * 64` is computed as signed `int`. Once `numBlocksX` is large enough, the multiplication wraps, and `rowBlock[comp]` points backward rather than forward into the temporary decode buffer.

Later, `LossyDctDecoder_execute()` uses those derived pointers for real loads and stores during the block shuffle and reconstruction process. At that point the decoder is no longer operating within the bounds of the allocation created for `rowBlockHandle`.

The public control flow is the standard one:

```
InputFile / ScanLineInputFile public read
  -> exr_decoding_run(...)
     -> exr_uncompress_chunk(...)
        -> internal_exr_undo_dwaa(...)
           -> DwaCompressor_uncompress(...)
              -> LossyDctDecoder_execute(...)
```

UBSan gives a clean root-cause diagnosis on the overflowing multiply, while ASAN shows the later memory error in the write-side decode path.

Reproduction

dwa_scanline_exrcheck.zip

Build with `exrcheck` with ASAN and run:

```
❯ ./build-asan/bin/exrcheck /tmp/dwa_scanline_exrcheck.exr
 file /tmp/dwa_scanline_exrcheck.exr
/home/pop/sec/openexr/src/lib/OpenEXRCore/internal_dwa_decoder.h:331:58: runtime
error: signed integer overflow: 33554432 * 64 cannot be represented in type 'int'
AddressSanitizer:DEADLYSIGNAL
=================================================================
==1684058==ERROR: AddressSanitizer: SEGV on unknown address 0x758f8e5f0800 (pc
0x75979e850336 bp 0x7ffe8f1d3420 sp 0x7ffe8f1d30f0 T0)
==1684058==The signal is caused by a WRITE memory access.
    #0 0x75979e850336 in LossyDctDecoder_execute
/home/pop/sec/openexr/src/lib/OpenEXRCore/internal_dwa_decoder.h:524
    #1 0x75979e879592 in DwaCompressor_uncompress
/home/pop/sec/openexr/src/lib/OpenEXRCore/internal_dwa_compressor.h:1210
    #2 0x75979e879592 in internal_exr_undo_dwaa
/home/pop/sec/openexr/src/lib/OpenEXRCore/internal_dwa.c:231
    #3 0x75979e95f878 in exr_uncompress_chunk
/home/pop/sec/openexr/src/lib/OpenEXRCore/compression.c:542
    #4 0x75979e9659a8 in exr_decoding_run
/home/pop/sec/openexr/src/lib/OpenEXRCore/decoding.c:580
    #5 0x7597a0271add in run_decode
/home/pop/sec/openexr/src/lib/OpenEXR/ImfScanLineInputFile.cpp:586
    #6 0x7597a0283dc4 in
Imf_4_0::ScanLineInputFile::Data::readPixels(Imf_4_0::FrameBuffer const&, int, int)
/home/pop/sec/openexr/src/lib/OpenEXR/ImfScanLineInputFile.cpp:500
    #7 0x7597a00c6a81 in Imf_4_0::InputFile::Data::readPixels(int, int)
/home/pop/sec/openexr/src/lib/OpenEXR/ImfInputFile.cpp:458
    #8 0x7597a13fe2dc in readScanline<Imf_4_0::InputPart>
/home/pop/sec/openexr/src/lib/OpenEXRUtil/ImfCheckFile.cpp:239
    #9 0x7597a1405b04 in readMultiPart
```

```
/home/pop/sec/openexr/src/lib/OpenEXRUtil/ImfCheckFile.cpp:905
    #10 0x7597a14126fd in runChecks<char const*>
/home/pop/sec/openexr/src/lib/OpenEXRUtil/ImfCheckFile.cpp:1171
    #11 0x7597a14146b9 in Imf_4_0::checkOpenEXRFile(char const*, bool, bool, bool)
/home/pop/sec/openexr/src/lib/OpenEXRUtil/ImfCheckFile.cpp:1835
    #12 0x61ba9582b8f8 in exrCheck(char const*, bool, bool, bool, bool)
/home/pop/sec/openexr/src/bin/exrcheck/main.cpp:96
    #13 0x61ba958282b1 in main /home/pop/sec/openexr/src/bin/exrcheck/main.cpp:164
    #14 0x75979d62a1c9 in __libc_start_call_main
../sysdeps/nptl/libc_start_call_main.h:58
    #15 0x75979d62a28a in __libc_start_main_impl ../csu/libc-start.c:360
    #16 0x61ba95829844 in _start (/home/pop/sec/openexr/build-
asan/bin/exrcheck+0xe844) (BuildId: 087c972343a5372940c42c0a2e7bce4a84288aec)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV
/home/pop/sec/openexr/src/lib/OpenEXRCore/internal_dwa_decoder.h:524 in
LossyDctDecoder_execute
==1684058==ABORTING
```

Found by: Quang Luong of Calif.io

Severity

High 8.4 / 10

CVSS v4 base metrics

Exploitability Metrics

Attack Vector	Local
Attack Complexity	Low
Attack Requirements	None
Privileges Required	None
User interaction	Active

Vulnerable System Impact Metrics

Confidentiality	High
Integrity	High
Availability	High

Subsequent System Impact Metrics

Confidentiality	None
Integrity	None
Availability	None

Learn more about base metrics

CVSS:4.0/AV:L/AC:L/AT:N/PR:N/UI:A/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N

CVE ID

CVE-2026-34589

Weaknesses

- CWE-190
- CWE-787

Credits

quangIO Reporter