huntr

chafa <= 4bac1466 is vulnerable to an out of bounds read vulnerability. in hpjansson/chafa

0

✓ Valid Reported on Jun 7th 2022

chafa <= 4bac1466 is vulnerable to an out of bounds read vulnerability.

Building

Build chafa with ASAN(address sanitizer)

```
$ git rev-parse HEAD
4bac14668535c09f6f47552bbd1566097dab4bf8
$ export CFLAGS="-g -O0 -fsanitize=address"; export CXXFLAGS="-g -O0 -fsani
$ ./autogen.sh
$ ./configure --disable-shared
$ make -j 8
```

Reproduction

PoC available here - google-drive

```
$ ./tools/chafa/chafa /tmp/cc6d16b6c395925244b398c849a02a47625f67dc
=================================================================
==4615==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62a00000
READ of size 1 at 0x62a000005259 thread T0
    #0 0x5abae5 in lzw_decode /tmp/chafa/libnsgif/lzw.c:362:45
    #1 0x5a7901 in gif_internal_decode_frame /tmp/chafa/libnsgif/libnsgif.c
    #2 0x5a541c in gif_decode_frame /tmp/chafa/libnsgif/libnsgif.c:1151:16
    #3 0x4d89c7 in maybe_decode_frame /tmp/chafa/tools/chafa/gif-loader.c:6
    #4 0x4d8569 in gif_loader_get_frame_data /tmp/chafa/tools/chafa/gif-loa
    #5 0x4d9b47 in media_loader_get_frame_data /tmp/chafa/t
    #6 0x4cea97 in run_generic /tmp/chafa/tools/chafa/chafa.
    #7 0x4ce3a6 in run /tmp/chafa/tools/chafa/chafa.c:1923:12
```

Chat with us

```
    #8 0x4c74a4 in run_all /tmp/chafa/tools/chafa/chafa.c:1983:18
    #9 0x4c5dd1 in main /tmp/chafa/tools/chafa/chafa.c:2035:11
    #10 0x7fc448ca9c86 in __libc_start_main /build/glibc-CVJwZb/glibc-2.27/

    #11 0x41ddb9 in _start (/tmp/chafa/tools/chafa/chafa+0x41ddb9)

0x62a000005259 is located 9 bytes to the right of 20560-byte region [0x62a
allocated by thread T0 here:
    #0 0x4964fd in malloc (/tmp/chafa/tools/chafa/chafa+0x4964fd)
    #1 0x5a9f05 in lzw_context_create /tmp/chafa/libnsgif/lzw.c:91:22
    #2 0x5a0823 in gif_initialise /tmp/chafa/libnsgif/libnsgif.c:946:34
    #3 0x4d7e21 in gif_loader_new_from_mapping /tmp/chafa/tools/chafa/gif-l
    #4 0x4d93ba in media_loader_new /tmp/chafa/tools/chafa/media-loader.c:
    #5 0x4ce8cf in run_generic /tmp/chafa/tools/chafa/chafa.c:1747:20
    #6 0x4ce3a6 in run /tmp/chafa/tools/chafa/chafa.c:1923:12
    #7 0x4c74a4 in run_all /tmp/chafa/tools/chafa/chafa.c:1983:18
    #8 0x4c5dd1 in main /tmp/chafa/tools/chafa/chafa.c:2035:11
    #9 0x7fc448ca9c86 in __libc_start_main /build/glibc-CVJwZb/glibc-2.27/

SUMMARY: AddressSanitizer: heap-buffer-overflow /tmp/chafa/libnsgif/lzw.c:
Shadow bytes around the buggy address:
  0x0c547fff89f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c547fff8a40: 00 00 00 00 00 00 00 00 00 00 fa[fa]fa fa fa fa
  0x0c547fff8a50: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a60: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a70: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a80: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a90: 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
```

Chat with us

```
  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
==4615==ABORTING
```

Analysis

I know that the stacktrace shows `libnsgif` which has upstream - https://github.com/netsurf-plan9/libnsgif but this is the only downstream project that has a good security posture. Also the upsteam doesn't build that well with missing instructions. I'll try to build and update here in a later comment.
The `table` is allocated with a fixed size of `1 << LZW_CODE_MAX` = 0x1000 members. But here in the PoC it could be larger

```
gef➤  print (code >> current_bit) & ((1 << code_size) - 1)
$5 = 0x1002
```

This would read outof the bounds for this array. I haven't looked into much on why we the values get to this state. This PoC is from a fuzzing session.
Thanks

Impact

The vulnerability by itself doesn't seem to have much impact. It can only read out of the bounds and leak some values from the heap. This can only be powerful once paired with any other primitive which can write.
I only report here as I see this project active and the upstream with no activity. If the project owner feels I can defer the report to that project too.

Chat with us

CVE

CVE-2022-2061
(Published)

Vulnerability Type
CWE-122: Heap-based Buffer Overflow

Severity
Low (2.8)

Registry
Other

Affected Version
<= 4bac1466

Visibility
Public

Status
Fixed

Found by

Sudhakar Verma
@sudhackar
unranked

Fixed by

Sudhakar Verma
@sudhackar
unranked

This report was seen 544 times.

We are processing your report and will contact the **hpjansson/chafa** team within 24 hours.
6 months ago

Sudhakar Verma modified the report 6 months ago

Sudhakar Verma 6 months ago

Researcher

Chat with us

The stacktrace can be a off by one line in as I added some debugging in libns
verified the PoC to work on commit 4bac14668535c09f6f47552bbd1566097dab4bf8

Sudhakar Verma modified the report 6 months ago

Sudhakar Verma 6 months ago Researcher

never mind. I fixed the asan trace.

Sudhakar Verma 6 months ago Researcher

This is reproducible upstream too. Here's how to repro upstream
I used the example tool - from here
with this change

```
#include <libnsgif.h>
```

instead of

```
#include "../include/libnsgif.h"
```

saving it as example.c in the chafa project root
if the build from last repro is still not cleaned just use

```
$ clang-10 -g -O0 -fsanitize=address example.c -I ./libnsgif ./libnsgif/.libs/libnsgif
$ ./example /tmp/cc6d16b6c395925244b398c849a02a47625f67dc
P3
# /tmp/cc6d16b6c395925244b398c849a02a47625f67dc
# width                10
# height               10
# frame_count          1
# frame_count_partial  1
# loop_count           1
10 10 256
=================================================================
==30953==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62a000005259 at pc
READ of size 1 at 0x62a000005259 thread T0
    #0 0x4cf9fb in lzw_decode /tmp/chafa/libnsgif/lzw.c:362:45
    #1 0x4cb881 in gif_internal_decode_frame /tmp/chafa/libnsgif/libnsgif.c:868:47
```

Chat with us

```
    #3 0x4c4205 in write_ppm /tmp/chafa/example.c:160:24
    #4 0x4c33c7 in main /tmp/chafa/example.c:239:9
    #5 0x7f5ab5fcbc86 in __libc_start_main /build/glibc-CVJwZb/glibc-2.27/csu/../csu/l

    #6 0x41aee9 in _start (/tmp/chafa/example+0x41aee9)

0x62a000005259 is located 9 bytes to the right of 20560-byte region [0x62a000000200,0x
allocated by thread T0 here:
    #0 0x49362d in malloc (/tmp/chafa/example+0x49362d)
    #1 0x4cde85 in lzw_context_create /tmp/chafa/libnsgif/lzw.c:91:22
    #2 0x4c47a3 in gif_initialise /tmp/chafa/libnsgif/libnsgif.c:946:34
    #3 0x4c32f3 in main /tmp/chafa/example.c:230:24
    #4 0x7f5ab5fcbc86 in __libc_start_main /build/glibc-CVJwZb/glibc-2.27/csu/../csu/l

SUMMARY: AddressSanitizer: heap-buffer-overflow /tmp/chafa/libnsgif/lzw.c:362:45 in lz
Shadow bytes around the buggy address:
  0x0c547fff89f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c547fff8a30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c547fff8a40: 00 00 00 00 00 00 00 00 00 00 fa[fa]fa fa fa fa
  0x0c547fff8a50: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a60: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a70: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a80: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c547fff8a90: 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
==30953==ABORTING
```

Chat with us

Sudhakar Verma 6 months ago Researcher

Oh I just realized its located someplace else - upstream

```
$ git clone git://git.netsurf-browser.org/libnsgif.git
```

Sudhakar Verma 6 months ago Researcher

And it is not vulnerable. You probably need to sync with a better updated version.
What I tried

```
[test] git rev-parse HEAD
a1c436270011cb3770f3d74bf08347289feb1273
[test] git remote get-url --all origin
git://git.netsurf-browser.org/libnsgif.git
[libnsgif] cd src
[src] clang-10 -g -O0 -fsanitize=address -std=c99 -shared -fPIC -I ../include gif.c lz
[src] cd ..
[libnsgif] cd test
[test] clang-10 -g -O0 -fsanitize=address -std=c99 -I ../include nsgif.c ../src/gif.a
[test] ./nsgif
Insufficient positional arguments found.

Usage: ./nsgif <FILE> [options]

Where:

  FILE  Path to GIF file to load.

Options:

  -m  --ppm STRING  Convert frames to PPM image at given path.

  -i  --info        Dump GIF info to stdout.

  -l  --loops UINT  Loop through decoding all frames N times. The de

  -p  --palette     Save palette images.
```

Chat with us

```
[test] ./nsgif /tmp/cc6d16b6c395925244b398c849a02a47625f67dc


nsgif_data_scan failed: Invalid frame data
nsgif_frame_prepare failed: Frame can't be displayed
```

Sudhakar Verma 6 months ago Researcher

I can raise a PR once the maintainer approves this report. Thanks!

We have contacted a member of the **hpjansson/chafa** team and are waiting to hear back
6 months ago

Hans Petter Jansson validated this vulnerability 6 months ago

Hi, that's a great analysis! I've confirmed the issue here using Valgrind with your POC.

The reason Chafa ships with the 0.2.1 release of libnsgif is that there are no newer releases, and I don't want to pull in something that's half-done/in the middle of a rewrite. The upstream code has changed a lot.

Marking as valid and awaiting PR. Would you mind including the POC file too, adding it under the tests/data/bad/ directory?

Sudhakar Verma has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

Sudhakar Verma 6 months ago Researcher

Sure. Thanks.

Sudhakar Verma 6 months ago

Chat with us

I can just backport the patch for this vulnerability

Sudhakar Verma 6 months ago Researcher

Hi
I have raised a PR for this. Sadly I could not backport the patch as there was no specific patch.

https://github.com/hpjansson/chafa/pull/93

Hans Petter Jansson gave praise 6 months ago

Merged. Thanks a lot. Not sure if I have to mark it as fixed, or if you can do that. huntr seems to want a fixed release version number, so I guess it'll have to wait until I make a new release? Might be a week or so in that case.

The researcher's credibility has slightly increased as a result of the maintainer's thanks: +1

Sudhakar Verma 6 months ago Researcher

You can mark as fixed by putting in a commit hash as well.

Sudhakar Verma submitted a patch 6 months ago

Hans 6 months ago Maintainer

Looks like it says "version the fix has been applied to", which seems ambiguous (does it mean "version that shipped with the fix" or "version the patch was diffed against"?) Anyway, I'll put the previous release in that field to not drag this out unnecessarily.

Thanks again :)

Hans Petter Jansson marked this as fixed in 1.12.0 with commit e6ce37 6 months ago

Sudhakar Verma has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

Sudhakar Verma 6 months ago Researcher

@admin can we get a CVE for this? This project is distributed as packages in major linux distros - Ubuntu and such.

Chat with us

Also please credit to - Sudhakar Verma of Crowdstrike

Hans 5 months ago Maintainer

No objection to CVE assignment from me.

Jamie Slome 5 months ago Admin

CVE assigned 👍 It should be published shortly!

Sudhakar Verma 5 months ago Researcher

I see CVE-2022-2061. Thank You
https://github.com/CVEProject/cvelist/pull/6075

Sign in to join this conversation