Closed · Issue created 1 year ago by Doneinq

Heap-buffer-overflow in dissect_bthci_iso at packet-bthci_iso.c

Summary

In Wireshark-3.5.1rc0, the bthci_iso dissector could crash with a heap-based buffer overflow. This issue also exists in the latest version v3.7.0rc0.

Steps to reproduce

- The location of the bug in the code. At line **410** in file **packet-bthci_iso.c**, the fourth parameter `len` of `tvb_memcpy` is read from the data packet without length check. The heap size of the copy target `mfp->reassembled + mfp->cur_off` can be controlled.

```
405         else if (pb_flag & 0x01) { /* continuation/last fragment */
406             mfp = (multi_fragment_pdu_t *)wmem_tree_lookup32_le(chandle_data->start_fragments, pinfo->num);
407             if (!pinfo->fd->visited) {
408                 len = tvb_captured_length_remaining(tvb, offset);
409                 if (mfp != NULL && !mfp->last_frame) {
410                     tvb_memcpy(tvb, (guint8 *) mfp->reassembled + mfp->cur_off, offset, len);
411                     mfp->cur_off += len;
412                     if (pb_flag == 0x03)
413                         mfp->last_frame = pinfo->num;
414                 }
415             }
```

- The bug requires the construction of two data packets. When `pb_flag == 0x00` , insert the data of the first fragment by calling `wmem_tree_insert32(chandle_data->start_fragments, pinfo->num, mfp);` .

```
376         if (pb_flag == 0x00) { /* first fragment */
377             if (!pinfo->fd->visited) {
378                 gint timestamp_size = 0;
379                 mfp = (multi_fragment_pdu_t *) wmem_new(wmem_file_scope(), multi_fragment_pdu_t);
380                 mfp->first_frame = pinfo->num;
381                 mfp->last_frame  = 0;
382                 mfp->tot_len     = 4;
383                 len = tvb_captured_length_remaining(tvb, offset);
384                 if (flags & 0x4000) { /* 4 byte timestamp is present */
385                     timestamp_size = 4;
386                 }
387                 mfp->tot_len += timestamp_size + (tvb_get_letohs(tvb, offset + 2 + timestamp_size) & 0xfff);
388                 mfp->reassembled = (char *) wmem_alloc(wmem_file_scope(), mfp->tot_len);
389                 if (len <= mfp->tot_len) {
390                     tvb_memcpy(tvb, (guint8 *) mfp->reassembled, offset, len);
391                     mfp->cur_off = len;
392                     wmem_tree_insert32(chandle_data->start_fragments, pinfo->num, mfp);
393                 }
```

- Then, the size `mfp->tot_len` of the heap object `mfp->reassembled` can be controlled.
- Finally, the bug is triggered by the second packet when `pb_flag & 0x01` at line **410**.

What is the current bug behavior?

The bug can cause out-of-bounds memory reads and writes.

Relevant logs and/or screenshots

The Crash State with ASAN:

```
==================================================================
==fuzzshark-while==347==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x60f000005ea4 at pc 0x7ffff76232d3 bp 0x7ff    fffff26a0 sp 0x7fffffff1e48
WRITE of size 244 at 0x60f000005ea4 thread T0
    #0 0x7ffff76232d2 in __interceptor_memcpy ../../../../src/libsanitizer/sanitizer_common/sanitizer_common_interceptors.inc    :827
    #1 0x55555dd1f41 in dissect_bthci_iso ../epan/dissectors/packet-bthci_iso.c:410
    #2 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #3 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #4 0x55555d0ed1ad in dissector_try_uint_new ../epan/packet.c:1454
    #5 0x55555d9a549c in dissect_hci_h4 ../epan/dissectors/packet-hci_h4.c:97
    #6 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #7 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #8 0x55555d0ed1ad in dissector_try_uint_new ../epan/packet.c:1454
    #9 0x55555d430bc6 in dissect_bluetooth ../epan/dissectors/packet-bluetooth.c:3776
    #10 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #11 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #12 0x55555d0ed1ad in dissector_try_uint_new ../epan/packet.c:1454
    #13 0x55555dee0dab in dissect_pcap_pktdata ../epan/dissectors/packet-pcap_pktdata.c:378
    #14 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #15 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #16 0x55555d0f0586 in call_dissector_with_data ../epan/packet.c:3287
    #17 0x55555dff4ae3 in dissect_rftap ../epan/dissectors/packet-rftap.c:321
    #18 0x55555dff50bc in dissect_rftap_heur ../epan/dissectors/packet-rftap.c:342
    #19 0x55555d0ef476 in dissector_try_heuristic ../epan/packet.c:2868
    #20 0x55555e35a385 in decode_udp_ports ../epan/dissectors/packet-udp.c:712
    #21 0x55555e359b23 in dissect ../epan/dissectors/packet-udp.c:1267
    #22 0x55555e35d6f1 in dissect_udp ../epan/dissectors/packet-udp.c:1273
    #23 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #24 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #25 0x55555d0f5721 in call_all_postdissectors ../epan/packet.c:3649
    #26 0x55555d863a5d in dissect_frame ../epan/dissectors/packet-frame.c:1001
    #27 0x55555d0e8ad9 in call_dissector_through_handle ../epan/packet.c:761
    #28 0x55555d0ebd54 in call_dissector_work ../epan/packet.c:854
    #29 0x55555d0f0586 in call_dissector_with_data ../epan/packet.c:3287
    #30 0x55555d0f187f in dissect_record ../epan/packet.c:625
    #31 0x55555d0df876 in epan_dissect_run ../epan/epan.c:612
    #32 0x55555cfc31bc in LLVMFuzzerTestOneInput fuzz/fuzzshark-while.c:303
    #33 0x55555cfe34a1 in main fuzz/fuzzshark-while.c:1995
    #34 0x7ffff70b60b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
    #35 0x55555cfc2500 in _start (/work/victims/bins/aflasan/fuzzshark-while+0x7a6e50d)

0x60f000005ea4 is located 0 bytes to the right of 164-byte region [0x60f000005e00,0x60f000005ea4)
allocated by thread T0 here:
    #0 0x7ffff769dc47 in __interceptor_malloc ../../../../src/libsanitizer/asan/asan_malloc_linux.cpp:145
    #1 0x55555ede4358 in g_malloc (/work/victims/bins/aflasan/fuzzshark-while+0x9890358)
```

To upload designs, you'll need to enable LFS and have an admin enable hashed storage. More information

Tasks 0

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

Linked items 0

Link issues together to show that they're related or that one is blocking others. Learn more.

Related merge requests 3

- BT HCI_ISO: Don't overrun our reassembly buffer. !4603
- BT HCI_ISO: Don't overrun our reassembly buffer. !4607
- BT HCI_ISO: Don't overrun our reassembly buffer. !4608

When these merge requests are accepted, this issue will be closed automatically.

Activity

Gerald Combs @geraldcombs · 1 year ago — Owner

Hi @DoneinqCK, do you have a capture file that demonstrates this issue?

Doneinq @DoneinqCK · 1 year ago — Author

Hi @geraldcombs , I have attached the payload file.

Capture file

payload-vul1.pcap

Debug information

```
Command and args:  ./tshark -nVxr /work/payload-vul1.pcapng
```

```
(gdb)
410                         tvb_memcpy(tvb, (guint8 *) mfp->reassembled + mfp->cur_off, offset, len);
(gdb) p len
$8 = 1105
(gdb) p offset
$9 = 4
(gdb) p *tvb
value has been optimized out
(gdb) p *mfp
$10 = {first_frame = 1, last_frame = 0, tot_len = 254,
  reassembled = 0x7fffd641b1d0 "\004\005\372P    \377\377\377\377\377\377\377   \002 \377@  ", cur_off = 244}
```

```
(gdb) s
tvb_memcpy (tvb=0x61d000004320, target=0x7fffd641b2c4, offset=4, length=1105) at ../epan/tvbuff.c:855
855     {
(gdb) p *tvb
$11 = {next = 0x61d000004990, ops = 0x7ffff076a840 <tvb_subset_ops>, initialized = 1, flags = 0, ds_tvb = 0x61d00
  real_data = 0x61d00017492f "", length = 1109, reported_length = 1109, contained_length = 1109, raw_offset = 175
```

Gerald Combs mentioned in merge request !4603 (merged) 1 year ago

Gerald Combs closed via commit acaaad15 1 year ago

Gerald Combs mentioned in merge request !4607 (merged) 1 year ago

Gerald Combs mentioned in merge request !4608 (merged) 1 year ago

Doneing @DoneingCK · 1 year ago — Author

Hi @geraldcombs, can a CVE ID be assigned?

Gerald Combs @geraldcombs · 1 year ago — Owner

I typically do that one or two days prior to a release.

Doneing @DoneingCK · 1 year ago — Author

@geraldcombs Hi, please turn off the confidentiality. Thanks.

Gerald Combs @geraldcombs · 1 year ago — Owner

Hi @DoneingCK, how would you prefer to be credited?

Doneing @DoneingCK · 1 year ago — Author

Hi @geraldcombs , I want to get the CVE ID.

Gerald Combs @geraldcombs · 1 year ago — Owner

It's been assigned CVE-2021-39926. See also wnpa-sec-2021-12.

Doneing @DoneingCK · 1 year ago — Author

Hi @geraldcombs Thanks!

Please register or sign in to reply

Gerald Combs mentioned in commit da8a707b 1 year ago

Gerald Combs mentioned in commit 5877529f 1 year ago

Gerald Combs made the issue visible to everyone 1 year ago

Please register or sign in to reply