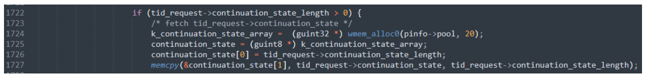
Closed Issue created 1 year ago by **Doneing**

Heap-buffer-overflow in reassemble_continuation_state at packet-btsdp.c

Summary

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

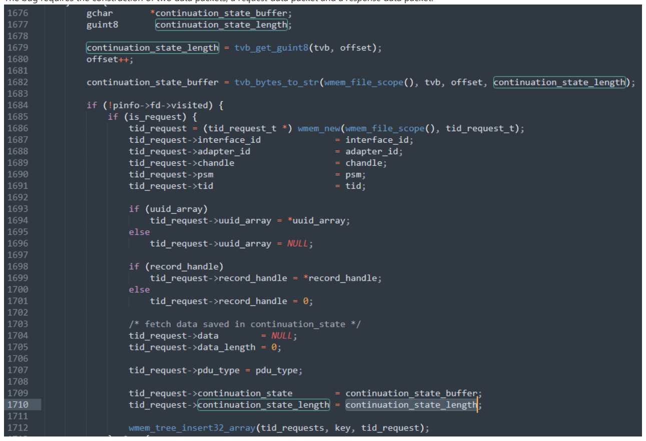
Steps to reproduce



```
1722                if (tid_request->continuation_state_length > 0) {
1723                    /* fetch tid_request->continuation_state */
1724                    k_continuation_state_array =  (guint32 *) wmem_alloc0(pinfo->pool, 20);
1725                    continuation_state = (guint8 *) k_continuation_state_array;
1726                    continuation_state[0] = tid_request->continuation_state_length;
1727                    memcpy(&continuation_state[1], tid_request->continuation_state, tid_request->continuation_state_length);
```

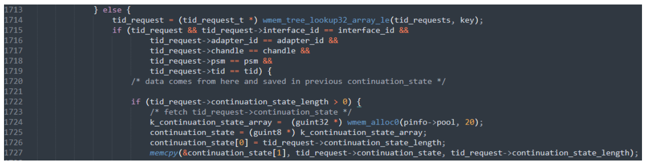
In line **1727**, the third parameter `tid_request->continuation_state_length` of memcpy is read from the data packet without length check.

The bug requires the construction of two data packets, a *request* data packet and a *response* data packet.



```
1676        gchar       *continuation_state_buffer;
1677        guint8       continuation_state_length;
1678
1679        continuation_state_length = tvb_get_guint8(tvb, offset);
1680        offset++;
1681
1682        continuation_state_buffer = tvb_bytes_to_str(wmem_file_scope(), tvb, offset, continuation_state_length);
1683
1684        if (!pinfo->fd->visited) {
1685            if (is_request) {
1686                tid_request = (tid_request_t *) wmem_new(wmem_file_scope(), tid_request_t);
1687                tid_request->interface_id              = interface_id;
1688                tid_request->adapter_id                = adapter_id;
1689                tid_request->chandle                   = chandle;
1690                tid_request->psm                       = psm;
1691                tid_request->tid                       = tid;
1692
1693                if (uuid_array)
1694                    tid_request->uuid_array = *uuid_array;
1695                else
1696                    tid_request->uuid_array = NULL;
1697
1698                if (record_handle)
1699                    tid_request->record_handle = *record_handle;
1700                else
1701                    tid_request->record_handle = 0;
1702
1703                /* fetch data saved in continuation_state */
1704                tid_request->data         = NULL;
1705                tid_request->data_length = 0;
1706
1707                tid_request->pdu_type = pdu_type;
1708
1709                tid_request->continuation_state        = continuation_state_buffer;
1710                tid_request->continuation_state_length = continuation_state_length;
1711
1712                wmem_tree_insert32_array(tid_requests, key, tid_request);
```

- First, the *request* packet inserts the object `tid_request` into the global object `tid_requests`. The field `tid_request->continuation_state_length` is read from the packet by `continuation_state_length = tvb_get_guint8(tvb, offset)`.



```
1713            } else {
1714                tid_request = (tid_request_t *) wmem_tree_lookup32_array_le(tid_requests, key);
1715                if (tid_request && tid_request->interface_id == interface_id &&
1716                        tid_request->adapter_id == adapter_id &&
1717                        tid_request->chandle == chandle &&
1718                        tid_request->psm == psm &&
1719                        tid_request->tid == tid) {
1720                /* data comes from here and saved in previous continuation_state */
1721
1722                if (tid_request->continuation_state_length > 0) {
1723                    /* fetch tid_request->continuation_state */
1724                    k_continuation_state_array =  (guint32 *) wmem_alloc0(pinfo->pool, 20);
1725                    continuation_state = (guint8 *) k_continuation_state_array;
1726                    continuation_state[0] = tid_request->continuation_state_length;
1727                    memcpy(&continuation_state[1], tid_request->continuation_state, tid_request->continuation_state_length);
```

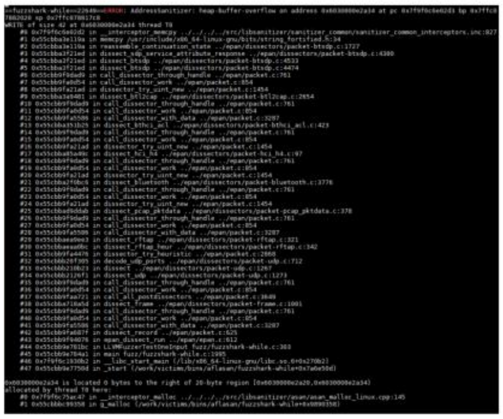
- Second, the *response* packet obtains the object `tid_request` by `wmem_tree_lookup32_array_le(tid_requests, key)`. When the value of variable `tid_request->continuation_state_length` is greater than 20, a heap overflow is caused.

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==22649==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6030000e2a34 at pc 0x7f9f6c6e02d3 bp 0x7ffc8
7882020 sp 0x7ffc87881 7c8
WRITE of size 42 at 0x6030000e2a34 thread T0
    #0 0x7f9f6c6e02d2 in __interceptor_memcpy ../../../../src/libsanitizer/sanitizer_common/sanitizer_common_interceptors.inc:827
    #1 0x55cbba3e119a in memcpy /usr/include/x86_64-linux-gnu/bits/string_fortified.h:34
    #2 0x55cbba3e119a in reassemble_continuation_state ../epan/dissectors/packet-btsdp.c:1727
    #3 0x55cbba3f21ed in dissect_sdp_service_attribute_response ../epan/dissectors/packet-btsdp.c:4300
    #4 0x55cbba3f21ed in dissect_btsdp ../epan/dissectors/packet-btsdp.c:4533
    #5 0x55cbba3f21ed in dissect_btsdp ../epan/dissectors/packet-btsdp.c:4474
    #6 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #7 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #8 0x55cbb9fa21ad in dissector_try_uint_new ../epan/packet.c:1454
    #9 0x55cbba3a9401 in dissect_btl2cap ../epan/dissectors/packet-btl2cap.c:2854
    #10 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #11 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #12 0x55cbb9fa5508 in call_dissector_with_data ../epan/packet.c:3287
    #13 0x55cbba351b25 in dissect_bthci_acl ../epan/dissectors/packet-bthci_acl.c:423
    #14 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #15 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #16 0x55cbb9fa21ad in dissector_try_uint_new ../epan/packet.c:1454
    #17 0x55cbba85a49c in dissect_hci_h4 ../epan/dissectors/packet-hci_h4.c:97
    #18 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #19 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #20 0x55cbb9fa21ad in dissector_try_uint_new ../epan/packet.c:1454
    #21 0x55cbba2f9bc6 in dissect_bluetooth ../epan/dissectors/packet-bluetooth.c:3776
    #22 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #23 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #24 0x55cbb9fa21ad in dissector_try_uint_new ../epan/packet.c:1454
    #25 0x55cbbad0ddab in dissect_pcap_pktdata ../epan/dissectors/packet-pcap_pktdata.c:378
    #26 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #27 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #28 0x55cbb9fa5508 in call_dissector_with_data ../epan/packet.c:3287
    #29 0x55cbbaea9ee3 in dissect_rftap ../epan/dissectors/packet-rftap.c:321
    #30 0x55cbbaeaa0bc in dissect_rftap_heur ../epan/dissectors/packet-rftap.c:342
    #31 0x55cbb9fa4476 in dissector_try_heuristic ../epan/packet.c:2868
    #32 0x55cbbb20f305 in decode_udp_ports ../epan/dissectors/packet-udp.c:712
    #33 0x55cbbb210b23 in dissect ../epan/dissectors/packet-udp.c:1267
    #34 0x55cbbb2126f1 in dissect_udp ../epan/dissectors/packet-udp.c:1273
    #35 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #36 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #37 0x55cbb9faa721 in call_all_postdissectors ../epan/packet.c:3649
    #38 0x55cbba718a5d in dissect_frame ../epan/dissectors/packet-frame.c:1001
    #39 0x55cbb9f9dad9 in call_dissector_through_handle ../epan/packet.c:761
    #40 0x55cbb9fa0d54 in call_dissector_work ../epan/packet.c:854
    #41 0x55cbb9fa5508 in call_dissector_with_data ../epan/packet.c:3287
    #42 0x55cbb9fa087f in dissect_record ../epan/packet.c:625
    #43 0x55cbb9f94076 in epan_dissect_run ../epan/epan.c:612
    #44 0x55cbb9e781bc in LLVMFuzzerTestOneInput fuzz/fuzzshark-while.c:388
    #45 0x55cbb9e784a1 in main fuzz/fuzzshark-while.c:1995
    #46 0x7f9f6c1030b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
    #47 0x55cbb9e7750d in _start (/work/victims/bins/aflasan/fuzzshark-while+0x7a0e50d)

0x6030000e2a34 is located 0 bytes to the right of 20-byte region [0x6030000e2a20,0x6030000e2a34)
allocated by thread T0 here:
    #0 0x7f9f6c75ac47 in __interceptor_malloc ../../../../src/libsanitizer/asan/asan_malloc_linux.cpp:145
    #1 0x55cbbbc90358 in g_malloc (/work/victims/bins/aflasan/fuzzshark-while+0x0090358)
```

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 4

BT SDP: Don't overrun our continuation state buffer.
!4604

BT SDP: Don't overrun our continuation state buffer.
!4844

BT SDP: Don't overrun our continuation state buffer.
!4845

BT SDP: Don't overrun our continuation state buffer.
!4846

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

Activity

Gerald Combs @geraldcombs · 1 year ago Owner

Hi @DoneingCK, thanks for the detailed writeup! Do you have a capture file that demonstrates this issue?

Gerald Combs mentioned in merge request !4604 (merged) 1 year ago

Doneing @DoneingCK · 1 year ago Author

Hi, @geraldcombs, this is a bug found by fuzzing interface `dissect_udp`. The data package is not complete. We are working hard to build a complete data package to demonstrate this issue.

Doneing @DoneingCK · 1 year ago Author

Hi @geraldcombs , I have attached the capture file.

Capture File

payload-vul2.pcapng

Debug information

```
Command and args:  ./tshark -nVxr payload-vul2.pcapng
```

```
Thread 1 "tshark" hit Breakpoint 3, reassemble_continuation_state (tvb=tvb@entry=0x555555619000,
    pinfo=pinfo@entry=0x5555556df968, offset=8, offset@entry=7, tid=tid@entry=8224, is_request=is_request@entry=0
    attribute_list_byte_offset=attribute_list_byte_offset@entry=7, attribute_list_byte_count=0, pdu_type=1,
    new_tvb=0x7fffffffbc28, is_first=0x7fffffffbc14, is_continued=0x7fffffffbc18, uuid_array=0x0, record_handle=0
    l2cap_data=<optimized out>, l2cap_data=<optimized out>) at ../epan/dissectors/packet-btsdp.c:1722
1722                            if (tid_request->continuation_state_length > 0) {
(gdb) n
1724                                    k_continuation_state_array =  (guint32 *) wmem_alloc0(pinfo->pool, 20);
(gdb)
1726                                    continuation_state[0] = tid_request->continuation_state_length;
(gdb)
1727                                    memcpy(&continuation_state[1], tid_request->continuation_state, tid_request->cont
(gdb) p tid_request->continuation_state_length
$1 = 42 '*'
(gdb)
```

- the size of destination buffer `&continuation_state[1]` of the memcpy is 19 bytes, and the length `tid_request->continuation_state_length` of the copy is 42 bytes.

Gerald Combs @geraldcombs · 1 year ago Owner

Thanks! I've verified the problem in master here using `WIRESHARK_DEBUG_WMEM_OVERRIDE=strict G_SLICE=debug-blocks lldb ./run/tshark`.

Doneing @DoneingCK · 1 year ago Author

@geraldcombs Hi, how to turn off the confidentiality?

Gerald Combs @geraldcombs · 1 year ago Owner

Done.

Doneing @DoneingCK · 1 year ago Author

Hi @geraldcombs , can I apply for a CVE ID for this vulnerability?

Gerald Combs @geraldcombs · 1 year ago Owner

It's been assigned CVE-2021-39925. See also wnpa-sec-2021-09.

Please register or sign in to reply

Gerald Combs closed via commit e15e9874 1 year ago

A Wireshark GitLab Utility closed via merge request !4604 (merged) 1 year ago

Gerald Combs mentioned in merge request !4844 (merged) 1 year ago

Gerald Combs mentioned in merge request !4845 (merged) 1 year ago

Gerald Combs mentioned in merge request !4846 (merged) 1 year ago

Doneing reopened 1 year ago

Doneing closed 1 year ago

Gerald Combs made the issue visible to everyone 1 year ago

Gerald Combs mentioned in commit 492a7038 1 year ago

Gerald Combs mentioned in commit b18691c5 1 year ago

Gerald Combs mentioned in commit ae45fcd6 1 year ago

Please register or sign in to reply