ESPILON
SECURITY RESEARCH

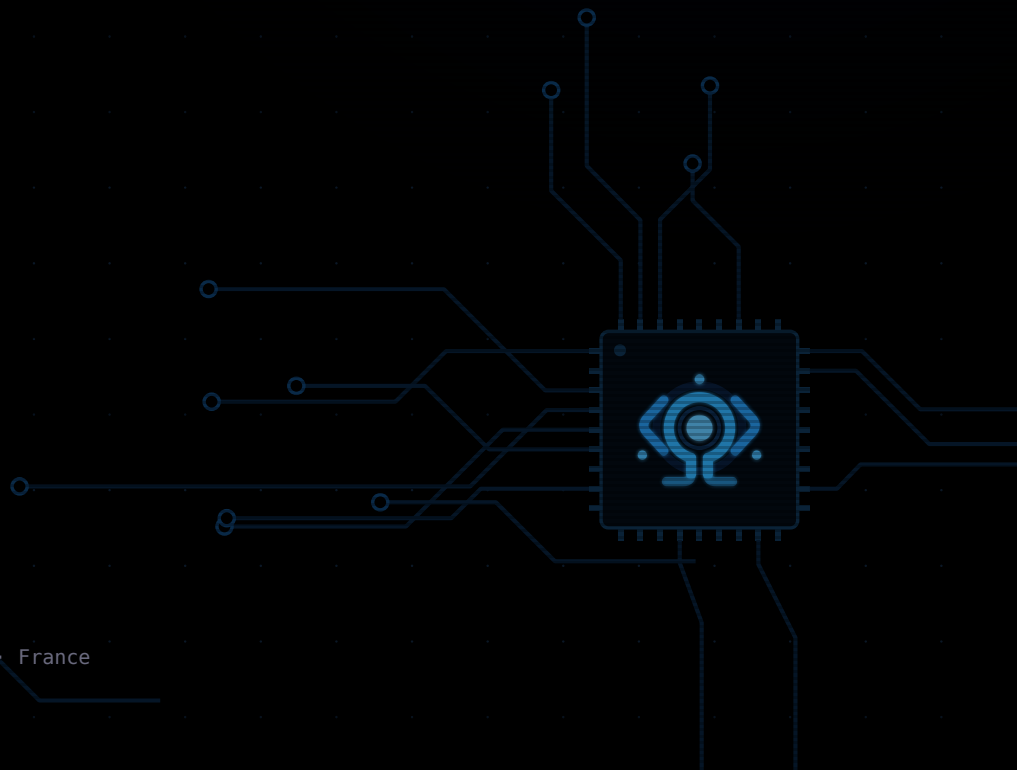
SECURITY ADVISORY

Heap out-of-bounds read in the Bluedroid AVRCP target parser

CVE-2026-46532 CVSS 4.6 · MEDIUM

Espressif · ESP-IDF · Bluedroid (AVRCP)

CVE PUBLISHED **10 Jun 2026**



Marouane El Moufid (Eun0us)

espilon.net · Association loi 1901 · France

01 Vulnerability

4.6

MEDIUM

CVSS 3.1 BASE SCORE · GITHUB (CNA), GHSA-3pp8-42fh-3j3c

AV:A / AC:L / PR:L / UI:N / S:U / C:L / I:N / A:L

CVE	CVE-2026-46532 (CNA: GitHub)
Advisory	GHSA-3pp8-42fh-3j3c, published 2026-05-19
Component	ESP-IDF Bluedroid, `avrc_pars_vendor_cmd()` in `components/bt/host/bluedroid/stack/avrc/avrc_pars_tg.c`
Type	Heap out-of-bounds read of one byte (CWE-125)
Attacker	Adjacent (Bluetooth range), paired BR/EDR peer with an AVRCP connection
Credit	Espilon · Marouane El Moufid (Eun0us)

Summary

In the Bluedroid AVRCP target, `avrc_pars_vendor_cmd()` parses the vendor-dependent commands sent by the connected controller. Two PDU handlers, `GET_CAPABILITIES` (0x10) and `LIST_PLAYER_APP_VALUES` (0x12), read the first parameter byte before checking that the command carries one. A command whose parameter length is zero makes the parser read one byte past the end of the received message. The value of that byte selects which of two AVRCP error codes the device sends back, which gives the peer a narrow oracle on adjacent heap memory. The byte itself is never transmitted.

Affected and fixed versions

The advisory lists the last vulnerable and the first patched release per branch; we checked each tag on 2026-10-28 for the fix. Builds without Classic Bluetooth AVRCP support (`CONFIG_BT_AVRCP_ENABLED`, selected by A2DP) are not affected.

BRANCH	AFFECTED	PATCHED	COMMIT	TAG CHECK (2026-10-28)
v5.2	5.2.6	5.2.7	c53d05ae	v5.2.7 carries the fix
v5.3	5.3.5	5.3.6	7c004d3	v5.3.6 carries the fix
v5.4	5.4.4	5.4.5	56053c4d	v5.4.5 not tagged yet; fix on `release/v5.4`
v5.5	5.5.3	5.5.4	b0959b5a	**v5.5.4 still has the vulnerable code**; first fixed tag is v5.5.5
v6.0	6.0	6.0.1	8746e5f7	v6.0.1 carries the fix
master			60f9362f	fixed on 2026-03-10

Timeline

DATE	EVENT
2026-03-10	Fix committed to ESP-IDF master (`60f9362f`); not yet in a release
2026-03-17	Reported by Espilon to the Espressif bug bounty programme, found on v5.5.3
2026-05-12	Espressif bug bounty programme confirms report #419 as valid
2026-05-14	CVE-2026-46532 reserved by the GitHub CNA
2026-05-19	Advisory GHSA-3pp8-42fh-3j3c published
2026-06-10	CVE record published

02 Technical analysis

Vulnerable code

The function checks that the vendor payload holds at least the four-byte fixed header (PDU ID, reserved byte, 16-bit parameter length), then checks that the declared parameter length matches the payload. The two handlers then dereference `p` without looking at `len`:

components/bt/host/bluedroid/stack/avrc/avrc_pars_tg.c at c2768cf8, parent of the master fix (abridged)

```
55        /* Check the vendor data */
56        if (p_msg->vendor_len == 0) {
57            return AVRC_STS_NO_ERROR;
58        }
59        if ((p_msg->p_vendor_data == NULL) || (p_msg->vendor_len < AVRC_CMD_FIXED_SIZE)) {
60            return AVRC_STS_INTERNAL_ERR;
61        }
62
63        p = p_msg->p_vendor_data;
64        p_result->pdu = *p++;
...
71        p++; /* skip the reserved byte */
72        BE_STREAM_TO_UINT16 (len, p);
73        if ((len + 4) != (p_msg->vendor_len)) {
74            status = AVRC_STS_NOT_FOUND;
75        }
76
77        if (status != AVRC_STS_NO_ERROR) {
78            return status;
79        }
80
81        switch (p_result->pdu) {
82        case AVRC_PDU_GET_CAPABILITIES:         /* 0x10 */
83            p_result->get_caps.capability_id = *p++;
84            if (!AVRC_IS_VALID_CAP_ID(p_result->get_caps.capability_id)) {
85                status = AVRC_STS_BAD_PARAM;
86            } else if (len != 1) {
87                status = AVRC_STS_INTERNAL_ERR;
88            }
89            break;
...
98        case AVRC_PDU_LIST_PLAYER_APP_VALUES:   /* 0x12 */
99            p_result->list_app_values.attr_id = *p++;
100           if (!AVRC_IS_VALID_ATTRIBUTE(p_result->list_app_values.attr_id)) {
101               status = AVRC_STS_BAD_PARAM;
102           } else if (len != 1) {
103               status = AVRC_STS_INTERNAL_ERR;
104           }
105           break;
```

Root cause

A vendor payload of exactly four bytes with a parameter length of 0 passes both checks: `vendor_len` is not below `AVRC_CMD_FIXED_SIZE` (4), and `len + 4` equals `vendor_len`. After line 72, `p` points at the first parameter byte, which for a zero-length command lies just past the payload. For PDU 0x10, line 83 reads that byte into `capability_id` for PDU 0x12, line 99 reads it into `attr_id`. The length test that would reject the command (`len != 1`) runs only after the read, and only on the branch where the byte passed the validity check.

What the peer learns

The order of the checks turns the stray byte into an observable result. For `GET_CAPABILITIES`, a byte equal to 0x02 or 0x03 (`AVRC_IS_VALID_CAP_ID`) yields `AVRC_STS_INTERNAL_ERR` (0x03), any other value yields `AVRC_STS_BAD_PARAM` (0x01). For `LIST_PLAYER_APP_VALUES`, the split is on `AVRC_IS_VALID_ATTRIBUTE` (1 to 4, or 0x80 and above). According to the advisory, the status code is carried in the AVRCP error response sent back to the peer, so one command reveals whether one adjacent heap byte belongs to a small set of values. Repeated over sessions this is a low-bandwidth signal, not a memory dump.

Reachability

The parser runs on the AVRCP target role, which ESP-IDF enables with `CONFIG_BT_AVRCP_ENABLED` (selected automatically by A2DP) on every Espressif chip with BR/EDR. Per the advisory, Bluedroid registers the AVCTP L2CAP service with `BTA_SEC_AUTHENTICATE`, so the sender must hold a valid BR/EDR link key from a previous pairing: a paired speaker, headset or controller that has an A2DP/AVRCP connection. This is not reachable by an unpaired device.

Impact

The official assessment is limited information disclosure about the heap memory next to the AVRCP receive buffer, and, in rare cases, a crash caused by the out-of-bounds access (`C:L/A:L`). Code execution has not been demonstrated and we make no such claim.

Evidence

Found by our libFuzzer campaign on the AVRCP target parser, reported to the Espressif bug bounty programme on 2026-03-17 and confirmed by Espressif as report #419. The harness hands `AVRC_ParsCommand()` a vendor payload copied into a heap block of exactly `vendor_len` bytes. The two crashing inputs kept from that campaign were replayed on 2026-10-28 with the campaign's binary, built on 2026-03-24; the lines it reports match the vulnerable source quoted above. Each is a 4-byte payload with a parameter length of 0, after two bytes the harness uses to pick the message type: `12 00 00 00` (PDU 0x12) and `10 01 00 00` (PDU 0x10).

• AddressSanitizer, fuzz_bt_avrc_tg, input ca 01 | 12 00 00 00 (replayed 2026-10-28)

```
==2759080==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000000054
READ of size 1 at 0x602000000054 thread T0
    #0 in avrc_pars_vendor_cmd  components/bt/host/bluedroid/stack/avrc/avrc_pars_tg.c:99:45
    #1 in AVRC_ParsCommand      components/bt/host/bluedroid/stack/avrc/avrc_pars_tg.c:300:22
    #2 in fuzz_avrc_tg          targets/bt_avrc_tg/fuzz_bt_avrc_tg.c:52:9
0x602000000054 is located 0 bytes to the right of 4-byte region [0x602000000050,0x602000000054)
SUMMARY: AddressSanitizer: heap-buffer-overflow avrc_pars_tg.c:99:45 in avrc_pars_vendor_cmd
```

The second input stops at `avrc_pars_tg.c:83:44`, the `GET_CAPABILITIES` read, with the same 1-byte read just past a 4-byte region. On a device the payload sits inside the larger AVCTP receive buffer, so the byte read lies beyond the message but usually still inside that buffer. This is the "adjacent heap memory" of the advisory, and the reason a crash is the rare case.

03 Fix and recommendation

Upstream fix

Commit `60f9362f` (Jin Cheng, Espressif, 2026-03-10, "fix(bt/bluedroid): fixed possible OOB read in `avrc_pars_vendor_cmd`") rejects a zero-length payload with `AVRC_STS_INTERNAL_ERR` before the byte is read, in both handlers. Once the read is gone, the status code no longer depends on memory outside the message. Full diff, verbatim:

• espressif/esp-idf commit 60f9362f83a05942069532f357c234cd5e5d4302, avrc_pars_tg.c

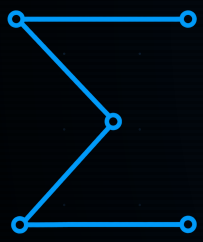
```
@@ -80,11 +80,13 @@ static tAVRC_STS avrc_pars_vendor_cmd(tAVRC_MSG_VENDOR *p_msg, tAVRC_COMMAND *p_

     switch (p_result->pdu) {
     case AVRC_PDU_GET_CAPABILITIES:         /* 0x10 */
-        p_result->get_caps.capability_id = *p++;
-        if (!AVRC_IS_VALID_CAP_ID(p_result->get_caps.capability_id)) {
-            status = AVRC_STS_BAD_PARAM;
-        } else if (len != 1) {
+        if (len < 1) {
             status = AVRC_STS_INTERNAL_ERR;
+        } else {
+            p_result->get_caps.capability_id = *p++;
+            if (!AVRC_IS_VALID_CAP_ID(p_result->get_caps.capability_id)) {
+                status = AVRC_STS_BAD_PARAM;
+            }
         }
         break;

@@ -96,11 +98,13 @@ static tAVRC_STS avrc_pars_vendor_cmd(tAVRC_MSG_VENDOR *p_msg, tAVRC_COMMAND *p_
         break;

     case AVRC_PDU_LIST_PLAYER_APP_VALUES:   /* 0x12 */
-        p_result->list_app_values.attr_id = *p++;
-        if (!AVRC_IS_VALID_ATTRIBUTE(p_result->list_app_values.attr_id)) {
-            status = AVRC_STS_BAD_PARAM;
-        } else if (len != 1) {
+        if (len < 1) {
             status = AVRC_STS_INTERNAL_ERR;
+        } else {
+            p_result->list_app_values.attr_id = *p++;
+            if (!AVRC_IS_VALID_ATTRIBUTE(p_result->list_app_values.attr_id)) {
+                status = AVRC_STS_BAD_PARAM;
+            }
         }
         break;
```

Recommendation

Update to v5.2.7, v5.3.6, v5.5.5 or v6.0.1 (on v5.4, the head of `release/v5.4` until v5.4.5 is tagged), or apply the commit for your branch from section 01. On the v5.5 branch, note that v5.5.4, listed as patched in the advisory, does not contain the change; use v5.5.5. Devices that do not need Classic Bluetooth audio can remove the surface by disabling A2DP and AVRCP.

04 Credit and references

- Credit: Marouane El Moufid (Eun0us), Espilon; the advisory credits “Eun0us / Espilon Worker” as reporter
- Advisory: github.com/espressif/esp-idf/security/advisories/GHSA-3pp8-42fh-3j3c
- CVE record: www.cve.org/CVERecord?id=CVE-2026-46532
- Fix (master): github.com/espressif/esp-idf/commit/60f9362f83a05942069532f357c234cd5e5d4302



ESPILON

SECURITY RESEARCH

Embedded, IoT and OT security research

espilon.net

eun0us@espilon.net

Association loi 1901 · France