Security Advisory: BES2300 Bluetooth Firmware Vulnerabilities

AVCTP / L2CAP Layer Issues (1.3, 1.4, 1.8)

Publication Date: 2026-09-01 | Vendor: Bestechnic Co., Ltd. | Severity: Critical

Overview

Vendor	Bestechnic Co., Ltd.
Product	BES2300 Bluetooth Audio SoC Firmware
CVE IDs	CVE-2026-79376, CVE-2026-79377, CVE-2026-79378
Affected Versions	v3.x and earlier (2018–2019 releases, End-of-Life)
Fixed Versions	v5.0 / RC06 and later (released 2023-05-15)
Attack Vector	Adjacent Network (Bluetooth radio range)
Discoverer	Denis Goryushev, Positive Technologies
Vendor Acknowledgment	Confirmed

Three independent security vulnerabilities have been identified in the Bluetooth protocol stack of the Bestechnic BES2300 Bluetooth Audio SoC firmware (v3.x and earlier). These vulnerabilities reside in the AVCTP and L2CAP layers and may lead to buffer overflow, integer truncation, and memory corruption.

The vulnerabilities are addressed in v5.0 / RC06 (released 15 May 2023) through architectural redesign. Legacy versions (v3.x and earlier) have entered End-of-Life and will not receive backported fixes; the supported remediation is upgrade to v5.0 / RC06 or later.

Vulnerability 1.3: AVCTP rx_frag_buff Unbounded Write via Fragmentation

CVE ID: `CVE-2026-79377`

CWE: `CWE-120`, `CWE-787`

CVSS v3.1: `CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H` → **8.8 (Critical)**

Description

In the BES2300 Bluetooth audio SoC firmware (v3.x and earlier, published 2018–2019, now End-of-Life), the AVCTP (Audio/Video Control Transport Protocol) layer supports fragmentation of large AVRCP PDUs into multiple L2CAP packets. The reassembly handler

`_avrcp_indicate_process_data_ind()` uses a fixed-size array **`rx_frag_buff[672]`** to cache incoming data.

When a START fragment is received, data is copied into

`rx_frag_buff` at offset 12. For each subsequent CONTINUE fragment (each carrying approximately 669 bytes of payload), data is appended at `rx_frag_buff[offset]` and the offset is incremented —

without any bounds check on the accumulated offset and without any limit on the number of CONTINUE fragments.

Per AVCTP specification v1.6.3, the control channel permits up to 255 fragments of up to 670 bytes each (672-byte L2CAP MTU minus 2-byte header), giving a theoretical maximum of 255 × 670 = 170,850 bytes, which far exceeds the statically allocated buffer. An attacker within Bluetooth radio range can transmit a crafted sequence of CONTINUE fragments, causing a heap or stack buffer overflow, memory corruption, and potential remote code execution or denial of service.

Exploitation Details

- Attack Vector: Adjacent Network (Bluetooth radio range, ~10 meters for Class 2 devices).
- Complexity: Low. No pairing or authentication required.
- Impact: Remote code execution (RCE) or denial of service (system crash).

Mitigation

The redesigned architecture (v5.0 / RC06, released 15 May 2023) does not allocate a fixed buffer at START. Instead, it evaluates the declared total length and dynamically resizes the reassembly buffer as needed. Additionally, an empirical upper bound of

`672 × 3 = 2016 bytes` is enforced on total AVRCP packet size; packets exceeding this limit are discarded with a diagnostic trace.

Legacy versions are End-of-Life and will not receive backported binary patches. For legacy-branch consumers who cannot immediately upgrade, a reference remediation diff (

`patch_1.3_avctp.diff`) is provided that adds the bounds check via **`avrcp_check_frag_offset()`**: before each memcpy, the condition `current_offset + incoming_len <= BTIF_AVRCP_RX_FRAG_BUFF_SIZE (672)` is evaluated.

Vulnerability 1.4: Integer Truncation in L2CAP PDU Length Processing

CVE ID: `CVE-2026-79378`

CWE: `CWE-190`, `CWE-787`

CVSS v3.1: `CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H` → **8.8 (Critical)**

Description

In the legacy BES2300 Bluetooth audio SoC firmware (v3.x and earlier, 2018–2019, now End-of-Life), the function

`btm_acl_handle()` processes incoming L2CAP PDUs carried over HCI ACL fragments. The first fragment (HCI packet boundary flag = START) carries the full L2CAP PDU total length in its 16-bit header field.

The handler reads the length via

`pkt_len = ((struct l2cap_hdr *)data_p)->len + 4`, where the +4 accounts for the L2CAP header itself. On ARM platforms, this expression undergoes integer promotion to int during calculation,

and the result is then truncated back to 16 bits when assigned to the uint16 pkt_len variable (via the **UXTH** instruction).

When the declared L2CAP length is 0xFFFF, the computation 0xFFFF + 4 = 0x10003 is truncated to 0x0003.

The code then calls

ppb_alloc(3), allocating only 3 bytes. Subsequently, **ppb_put_data()** invokes `memcpy(ppb->tail, data, len)` to copy the full ACL payload without any bounds check that `ppb->tail + len <= ppb->end`. After the first fragment, `ppb->tail` already exceeds `ppb->end`, and the completion check `if (ppb->tail == ppb->end)` never evaluates true.

Exploitation Details

- Attack Vector: Adjacent Network (Bluetooth radio range).
- Complexity: Low. A single malicious packet can trigger the overflow without requiring continuous fragmentation.
- Impact: Heap corruption, potential remote code execution, or denial of service.

Mitigation

The redesigned architecture (v5.0 / RC06, released 15 May 2023) adds an explicit bounds check before

`ppb_alloc`: if **`pkt_len > MAX_L2CAP_PACKET_LEN (0x4000) || pkt_len < 4`**, the packet is discarded and any partially assembled buffer is cleaned up. The fix is implemented in `btm_handle_hcievent.c` with the **MAX_L2CAP_PACKET_LEN** define set to 0x4000.

Legacy versions are End-of-Life and will not receive backported binary patches; the supported remediation is upgrade to v5.0 / RC06 or later. For legacy-branch consumers who cannot immediately upgrade, a reference remediation diff (

patch_1.4_l2cap.diff) is provided.

Vulnerability 1.8: Missing L2CAP RX MTU Enforcement

CVE ID: CVE-2026-79376

CWE: CWE-20, CWE-1284

CVSS v3.1: `CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H` → **8.8 (Critical)**

Description

In the BES2300 Bluetooth firmware, the function

l2cap_handle_data() receives reassembled L2CAP SDUs and dispatches them to upper-layer channel callbacks (AVDTP, AVCTP, RFCOMM, etc.) without validating whether the inbound packet length exceeds the negotiated L2CAP MTU.

During L2CAP connection establishment, both peers exchange MTU values via Configuration Request/Response (e.g., AVDTP typically negotiates 895 bytes, AVCTP control channel 672 bytes).

Upper-layer protocols allocate buffers assuming incoming data will not exceed this negotiated MTU. However,

`l2cap_handle_data()` only checks whether the channel state is OPEN, then directly invokes `channel->callback` to deliver the PPB to the upper layer.

An attacker within Bluetooth radio range can send L2CAP packets up to the PPB inplace buffer ceiling of 1052 bytes after connection establishment, causing upper-layer protocol buffers (designed around the MTU size) to overflow.

Exploitation Details

- Attack Vector: Adjacent Network (Bluetooth radio range).
- Prerequisite: L2CAP channel must be established (device paired or allowing anonymous connection).
- Impact: Memory corruption, denial of service (DoS); worst case may lead to RCE depending on upper-layer buffer layout.

Mitigation

Both legacy and newer branches require the same remediation: before invoking the upper-layer callback, verify that

`(cfgout_flag & L2CAP_SIG_CFG_MTU_MASK)` confirms MTU negotiation has completed, and that **`ppb->len <= mtu_remote.mtu`**. Packets exceeding the negotiated MTU are discarded with a diagnostic trace. The fix is implemented in `l2cap.c` as follows:

```
if ((channel->cfgout.cfgout_flag & L2CAP_SIG_CFG_MTU_MASK) && ppb->len > channel->cfgout.mtu_remote.mtu) { ...
```

Reference remediation diffs:

• Legacy branch:

`patch_1.8_mtu.diff` (applies to BES2300 v3.x and earlier)

• Newer branches: equivalent MTU enforcement patch

Legacy versions are End-of-Life; consumers on legacy branches who cannot immediately upgrade should apply the patch. Consumers on newer branches should ensure the MTU enforcement patch is integrated. The fully corrected behavior is present in v5.0 / RC06 and later builds that include the MTU check.

Summary

Feature	Vulnerability 1.3	Vulnerability 1.4	Vulnerability 1.8
CVE ID	CVE-2026-79377	CVE-2026-79378	CVE-2026-79376
Root Cause	Buffer Overflow (no bounds check)	Integer Truncation (UXTH wrap)	Missing Input Validation
Function	avrcp_indicate_process_data_ind	btm_acl_handle()	l2cap_handle_data()
Buffer/Variable	rx_frag_buff[672]	pkt_len (uint16)	ppb->len vs MTU

Trigger	Crafted CONTINUE fragments	L2CAP len = 0xFFFF	Oversized L2CAP SDU
Impact	Heap overflow → RCE / DoS	Heap overflow → RCE / DoS	Buffer overflow → DoS / RCE
CVSS	8.8 (Critical)	8.8 (Critical)	8.8 (Critical)
Fixed in	v5.0 / RC06+	v5.0 / RC06+	v5.0 / RC06+ (with patch)

Recommended Actions

- **Upgrade to v5.0 / RC06 or later** — This is the primary supported remediation. The AVCTP, L2CAP, and SBC subsystems have been completely redesigned to eliminate all listed vulnerabilities.
- **For legacy users unable to upgrade immediately:** Apply the reference patches (patch_1.3_avctp.diff, patch_1.4_l2cap.diff, patch_1.8_mtu.diff) to add bounds checking, integer overflow protection, and MTU enforcement respectively.
- Restrict Bluetooth pairing to trusted devices only; disable discoverability when not in use.
- **Note:** Legacy versions (v3.x and earlier) have entered End-of-Life and will not receive backported fixes from Bestechnic.

Credit

These vulnerabilities were discovered by Denis Goryushev of Positive Technologies.

References

• **CVE-2026-79376:** https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2026-79376

• **CVE-2026-79377:** https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2026-79377

• **CVE-2026-79378:** https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2026-79378

• **Vendor Product Page:** https://www.bestechnic.com

This advisory was prepared by Bestechnic Co., Ltd. in coordination with the discoverer.