



ESPILON
SECURITY RESEARCH

SECURITY ADVISORY

Heap buffer overflow in ESP-IDF protocomm Security2 over Bluetooth LE

CVE-2026-45542

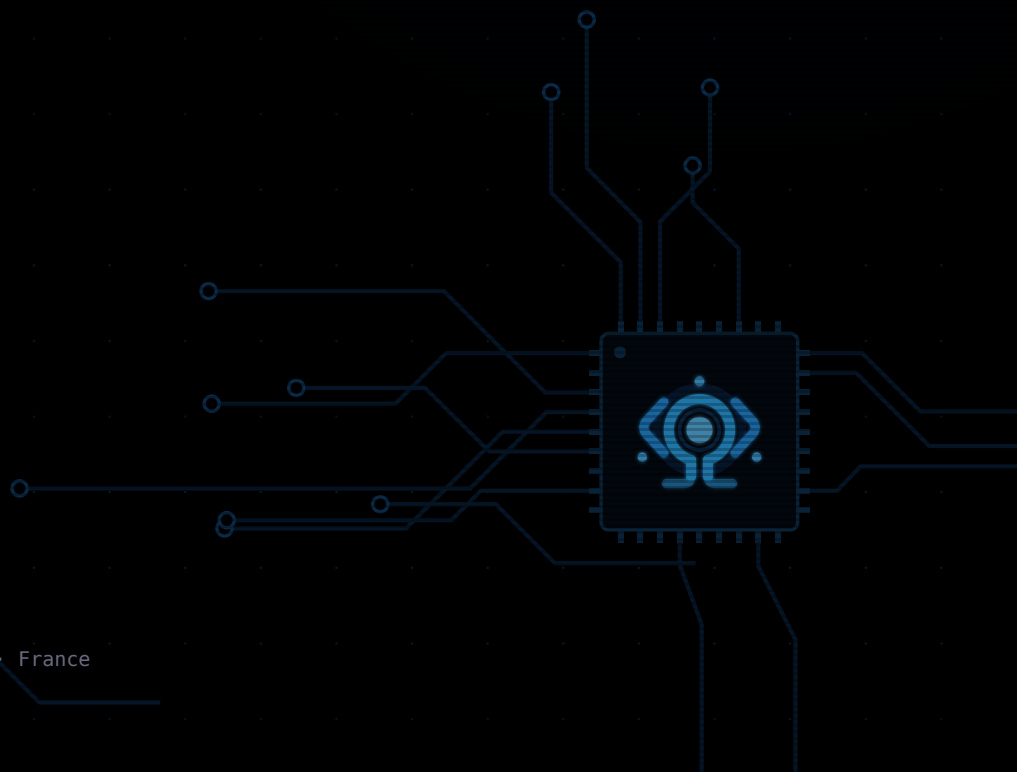
CVSS 7.1 · HIGH

Espressif · ESP-IDF · protocomm (Wi-Fi provisioning)

CVE PUBLISHED 10 Jun 2026

Marouane El Moufid (Eun0us)

espilon.net · Association loi 1901 · France



01 Vulnerability

7.1

CVSS 3.1 BASE SCORE · OFFICIAL (GHSA, GITHUB CNA)

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

HIGH

CVE	CVE-2026-45542 · GHSA-9r76-858f-v6jh
Product	Espressif ESP-IDF, component protocomm, Security Scheme 2 (SRP6a)
Location	components/protocomm/src/security/security2.c, handle_session_command0()
Type	Heap-based buffer overflow (CWE-122)
Attacker	Adjacent (BLE range), unauthenticated, before any SRP6a proof
Published	2026-05-18 (vendor advisory) · 2026-06-10 (CVE record)
Credit	Marouane El Moufid (Eun0us), Epsilon

Summary

Security Scheme 2 protects ESP-IDF provisioning sessions with SRP6a. The first message of the handshake carries the client's SRP6a username as a protobuf byte field. `handle_session_command0()` stores the field's length into a 16-bit structure member, allocates that many bytes, then copies the full, untruncated length. A username longer than 65535 bytes makes the allocation small and the copy large: the heap is overwritten with attacker bytes. The handler runs before any credential check, so no proof of possession is needed.

Affected and fixed versions

The vendor advisory lists the following releases as affected and fixed. Release dates and the presence of each branch commit in its fixed release were checked against the `esp-idf` repository on 2026-10-28.

BRANCH	AFFECTED	FIXED IN	FIX COMMIT
master			a2f4554
v6.0	v6.0	v6.0.1 (2026-04-27)	9b4cacf
v5.5	v5.5.4	v5.5.5 (2026-07-17)	71eb2db
v5.4	v5.4.4	v5.4.5, not released at the time of writing	f5d24a7
v5.3	v5.3.5	v5.3.6 (2026-09-15)	0ea58d7
v5.2	v5.2.6	v5.2.7 (2026-05-19)	56c3e38

Per the advisory, the flaw is reachable only when the application runs provisioning (or another `protocomm` service) over the NimBLE BLE transport with Security Scheme 2 (`PROTocomm_SEC_VER_2` / `WIFI_PROV_SECURITY_2`). The HTTP/SoftAP, Bluedroid BLE and console transports bound the inbound payload size and cannot reach the truncation threshold.

02 Technical details

Vulnerable code

The session keeps the username length in a `uint16_t`:

```
• components/protocomm/src/security/security2.c · ESP-IDF v6.0

51 typedef struct session {
52     /* Session data */
53     uint32_t id;
54     uint8_t state;
55     /* Currently fixing the salt length to 16, we may keep it flexible */
56     char *username;
57     uint16_t username_len;
```

The first-phase handler, with unrelated lines elided. The protobuf field `client_username.len` is a `size_t` the only bound is the test for zero on line 106.

```
• components/protocomm/src/security/security2.c · ESP-IDF v6.0 · handle_session_command0() (abridged)

84 static esp_err_t handle_session_command0(session_t *cur_session,
85     uint32_t session_id,
86     SessionData *req, SessionData *resp,
87     const protocomm_security2_params_t *sv)
88 {
89     ESP_LOGD(TAG, "Request to handle setup0_command");
90     Sec2Payload *in = (Sec2Payload *) req->Sec2;
91     [...]
98     if (in->sc0->client_pubkey.len != PUBLIC_KEY_LEN) {
99     [...]
106     if (in->sc0->client_username.len <= 0) {
107         ESP_LOGE(TAG, "Invalid username");
108     [...]
111     return ESP_ERR_INVALID_ARG;
112     }
113     [...]
189     cur_session->username_len = in->sc0->client_username.len;
190     cur_session->username = malloc(cur_session->username_len);
191     if (!cur_session->username) {
192         ESP_LOGE(TAG, "Failed to allocate memory!");
193         esp_srp_free(cur_session->srp_hd);
194         free(out);
195         free(out_resp);
196         return ESP_ERR_NO_MEM;
197     }
198     memcpy(cur_session->username, in->sc0->client_username.data, in->sc0->client_username.len);
```

Root cause

Line 189 narrows the wire length to 16 bits. For a username of 65536 + *n* bytes, `username_len` becomes *n*, line 190 allocates *n* bytes, and line 198 copies 65536 + *n* bytes into that allocation. The copy writes 65536 bytes past the end of the buffer, with content taken from the username field. The truncation-versus-copy asymmetry is the whole defect: the allocation and the copy use two different views of the same length.

What the attacker controls and reaches

The attacker chooses the allocation size *n* and every byte written. The overflow length is the part of the length lost in the narrowing, so it is a multiple of 65536 bytes. The only other field the handler checks first is the client public key, whose length must be `PUBLIC_KEY_LEN` (line 98); its content is passed to `esp_srp_get_session_key()` before the copy is reached. The SRP6a proof is verified only in the next message (`handle_session_command1()`), so the overflow happens before authentication. The attacker needs to be in BLE range of a device that has its provisioning service open.

03 Impact

The copy overwrites at least 64 KiB of heap beyond a small allocation, corrupting heap metadata and objects that belong to other tasks sharing the heap. The established consequence, as stated by the vendor, is denial of service: the device crashes and the provisioning service is lost. Code execution has not been demonstrated, and this paper does not claim it. The CNA vector reflects this: integrity low, availability high.

Proof

The vendor added a regression test with the fix. It builds exactly the input described above: a username of `UINT16_MAX + 2` (65537) bytes and a 384-byte public key, packed as a Security2 command 0 (packing elided) and handed to `protocomm_req_handle()`. On the vulnerable code this input gives `username_len = 1`, a one-byte allocation and a 65537-byte copy; on the fixed code the test asserts that the request is rejected. Log lines are elided.

• components/protocomm/test_apps/main/test_security2.c · added by a2f4554 · test_cmd0_oversized_username() (abridged)

```
94     const size_t username_len = (size_t)UINT16_MAX + 2;
95     uint8_t *username_buf = calloc(1, username_len);
[...]
```

```
101     memset(username_buf, 'A', username_len);
102
103     uint8_t client_pubkey[SEC2_PUBLIC_KEY_LEN];
104     memset(client_pubkey, 0xAB, sizeof(client_pubkey));
105
106     S2SessionCmd0 cmd0 = S2_SESSION_CMD0_INIT;
107     cmd0.client_username.data = username_buf;
108     cmd0.client_username.len = username_len;
109     cmd0.client_pubkey.data = client_pubkey;
110     cmd0.client_pubkey.len = SEC2_PUBLIC_KEY_LEN;
111
112     Sec2Payload payload = SEC2_PAYLOAD_INIT;
113     payload.msg = SEC2_MSG_TYPE_S2Session_Command0;
114     payload.payload_case = SEC2_PAYLOAD_PAYLOAD_SC0;
115     payload.sc0 = &cmd0;
[...]
```

```
135     ret = protocomm_req_handle(pc, "test-sec", session_id,
136                               packed, (ssize_t)packed_len,
137                               &outbuf, &outlen);
```

The defect was reported to Espressif through its bug bounty programme; Espressif fixed it, published the advisory and credited the report. The defect was found by code review of ESP-IDF v6.0-rc1 on 2026-03-21. The sanitizer run quoted in our report at the time exercised a reduced copy of the length handling, not the real `security2.c`, so it is not reproduced here.

04 Fix and recommendation

Upstream fix

The username length is checked against the range of the destination type before any allocation, and the copy uses the validated 16-bit length. The same commit also adds payload-type checks on both Security1 and Security2 handlers; those hunks are omitted below.

```
• 9b4cacf · release/v6.0 · components/protocomm/src/security/security2.c (abridged)

@@ -103,8 +108,8 @@ static esp_err_t handle_session_command0(session_t *cur_session,
     return ESP_ERR_INVALID_ARG;
 }

- if (in->sc0->client_username.len <= 0) {
-     ESP_LOGE(TAG, "Invalid username");
+ if (in->sc0->client_username.len == 0 || in->sc0->client_username.len > UINT16_MAX) {
+     ESP_LOGE(TAG, "Invalid username length (%zu)", in->sc0->client_username.len);
+     if (esp_event_post(PROTOCOMM_SECURITY_SESSION_EVENT,
PROTOCOMM_SECURITY_SESSION_INVALID_SECURITY_PARAMS, NULL, 0, portMAX_DELAY) != ESP_OK) {
         ESP_LOGE(TAG, "Failed to post secure session invalid security params event");
     }
 }

@@ -195,7 +200,7 @@ static esp_err_t handle_session_command0(session_t *cur_session,
     free(out_resp);
     return ESP_ERR_NO_MEM;
 }
- memcpy(cur_session->username, in->sc0->client_username.data, in->sc0->client_username.len);
+ memcpy(cur_session->username, in->sc0->client_username.data, cur_session->username_len);

resp->sec_ver = SEC_SCHEME_VERSION_SecScheme2;
resp->proto_case = SESSION_DATA__PROTO_SEC2;
```

Recommendation

Upgrade to ESP-IDF v6.0.1, v5.5.5, v5.3.6 or v5.2.7, or a later release of the same line. On the v5.4 line, no release carries the fix at the time of writing: cherry-pick f5d24a7 onto the product branch. Until then, the advisory suggests provisioning over SoftAP/HTTP where the product allows it, and opening BLE provisioning only during onboarding, stopping the transport as soon as the device is provisioned.

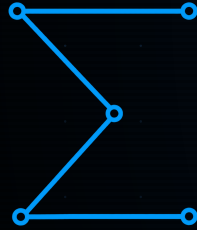
05 Timeline

DATE	EVENT
2026-04-01	Fix committed to master (a2f4554); release/v6.0 follows on 2026-04-07 (9b4cacf)
2026-04-27	ESP-IDF v6.0.1 released with the fix
2026-05-12	Espressif bug bounty reply: report accepted, advisory and CVE to follow
2026-05-18	Advisory GHSA-9r76-858f-v6jh published
2026-05-19	ESP-IDF v5.2.7 released with the fix
2026-06-10	CVE-2026-45542 record published (CNA: GitHub)
2026-07-17	ESP-IDF v5.5.5 released with the fix
2026-09-15	ESP-IDF v5.3.6 released with the fix

Credit and references

Credit: Marouane El Moufid (Eun0us), Espilon. The vendor advisory credits “Eun0us / Espilon Worker”.

- Vendor advisory: <https://github.com/espressif/esp-idf/security/advisories/GHSA-9r76-858f-v6jh>
- CVE record: <https://www.cve.org/CVERecord?id=CVE-2026-45542>
- Fix, master: [espressif/esp-idf@a2f4554](https://github.com/espressif/esp-idf/commit/a2f4554) · release/v6.0: [espressif/esp-idf@9b4cacf](https://github.com/espressif/esp-idf/commit/9b4cacf)



ESPILON

SECURITY RESEARCH

Embedded, IoT and OT security research

espilon.net

eun0us@espilon.net

Association loi 1901 • France