



**ESPILON**  
SECURITY RESEARCH

SECURITY ADVISORY

# Remote NULL pointer dereference in the ESP-IDF WebSocket server handshake

**CVE-2026-45541**

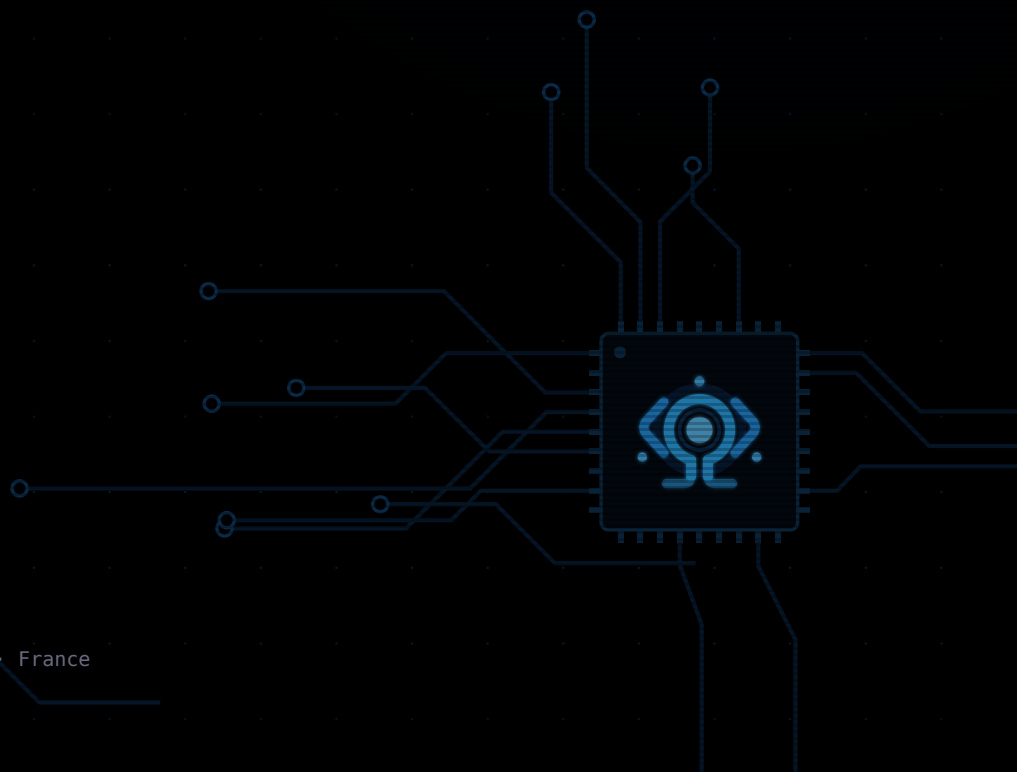
CVSS 7.5 · HIGH

Espressif · ESP-IDF · esp\_http\_server

CVE PUBLISHED 10 Jun 2026

Marouane El Moufid (Eun0us)

espilon.net · Association loi 1901 · France



# 01 Vulnerability

7.5

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

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

HIGH

|           |                                                                                |
|-----------|--------------------------------------------------------------------------------|
| CVE       | CVE-2026-45541 · <a href="#">GHSA-3j8v-xgrq-5vg8</a>                           |
| Product   | Espressif ESP-IDF, component esp_http_server (WebSocket support)               |
| Location  | components/esp_http_server/src/httpd_ws.c, httpd_ws_get_response_subprotocol() |
| Type      | NULL pointer dereference (CWE-476)                                             |
| Attacker  | Network, unauthenticated, no user interaction                                  |
| Published | 2026-05-15 (vendor advisory) · 2026-06-10 (CVE record)                         |
| Credit    | Marouane El Moufid (Eun0us), Epsilon                                           |

## Summary

The ESP-IDF HTTP server completes the WebSocket opening handshake itself, before it calls the application's URI handler. When the matched URI declares a subprotocol, the server splits the client's `Sec-WebSocket-Protocol` header with `strtok_r()` and passes the first token to `strncmp()` without checking it. A header value made only of separator characters, for example a single comma, yields no token: `strtok_r()` returns NULL and the server dereferences it. One unauthenticated HTTP request crashes the device.

## 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 |          |                                             | 9fc0ca1    |
| v6.0   | v6.0     | v6.0.1 (2026-04-27)                         | 0dc4ee7    |
| v5.5   | v5.5.4   | v5.5.5 (2026-07-17)                         | 00a2f7f    |
| v5.4   | v5.4.4   | v5.4.5, not released at the time of writing | 37508ab    |
| v5.3   | v5.3.5   | v5.3.6 (2026-09-15)                         | f88a47e    |
| v5.2   | v5.2.6   | v5.2.7 (2026-05-19)                         | dc46dc5    |

The vulnerable path exists only when the application is built with `CONFIG_HTTPD_WS_SUPPORT` and registers at least one WebSocket URI whose `supported_subprotocol` field is not NULL. The plain HTTP server and the HTTPS server (`esp_https_server`) are both affected, since TLS is terminated before the handshake is parsed.

## 02 Technical details

### Vulnerable code

The function below is quoted from the ESP-IDF v6.0 release. Lines 77 to 80 (the NULL check on the URI's own subprotocol and its log message) are elided.

• components/esp\_http\_server/src/httpd\_ws.c · ESP-IDF v6.0 (abridged)

```
70 static bool httpd_ws_get_response_subprotocol(const char *supported_subprotocol, char *subprotocol,
71 size_t buf_len)
72 {
73     /* Request didn't contain any subprotocols */
74     if (strlen(subprotocol, buf_len) == 0) {
75         return false;
76     }
77     [...]
81
82     /* Get first subprotocol from comma separated list */
83     char *rest = NULL;
84     char *s = strtok_r(subprotocol, ",", &rest);
85     do {
86         if (strncmp(s, supported_subprotocol, sizeof(subprotocol)) == 0) {
87             ESP_LOGD(TAG, "Requested subprotocol supported: %s", s);
88             return true;
89         }
90     } while ((s = strtok_r(NULL, ",", &rest)) != NULL);
91
92     ESP_LOGW(TAG, "Sec-WebSocket-Protocol %s not supported, supported subprotocol is %s",
93 subprotocol, supported_subprotocol);
94
95     /* No matches */
96     return false;
97 }
```

### Root cause

The emptiness test on line 73 runs before tokenisation and only rejects an empty string. A value such as , or , , passes it. `strtok_r()` skips leading delimiters; when nothing but delimiters remain it returns NULL. The loop is a `do { } while` whose condition is evaluated after the body, so the first iteration calls `strncmp(s, ...)` with `s == NULL` and reads address 0.

The comparison length on line 86 differs between releases: commit 4e9367bc41 on release/v6.0 replaced `sizeof(subprotocol)` with `strlen(supported_subprotocol)`, which is why the removed line in the fix diff below does not match line 86 of the v6.0 tag. The unchecked first token is the same in both.

### Where the input comes from

The header value is copied verbatim from the request into a 50-byte stack buffer, after the parser has skipped leading spaces, and handed to the function:

• components/esp\_http\_server/src/httpd\_ws.c · ESP-IDF v6.0 · `httpd_ws_respond_server_handshake()` (abridged)

```
175     char subprotocol[50] = { '\0' };
176     [...]
194     if ( httpd_ws_get_response_subprotocol(supported_subprotocol, subprotocol, sizeof(subprotocol)))
195     {
```

The handshake response is sent from the URI dispatcher, before the application's handler runs:

```
• components/esp_http_server/src/httpd_uri.c · ESP-IDF v6.0

322  /* Final step for a WebSocket handshake verification */
323  #ifdef CONFIG_HTTPD_WS_SUPPORT
324      struct httpd_req_aux *aux = req->aux;
325      if (uri->is_websocket && aux->ws_handshake_detect && uri->method == HTTP_GET) {
326  #ifdef CONFIG_HTTPD_WS_PRE_HANDSHAKE_CB_SUPPORT
327          if (uri->ws_pre_handshake_cb && uri->ws_pre_handshake_cb(req) != ESP_OK) {
328              ESP_LOGW(TAG, LOG_FMT("ws_pre_handshake_cb failed"));
329              return ESP_FAIL;
330          }
331  #endif
332      ESP_LOGD(TAG, LOG_FMT("Responding WS handshake to sock %d"), aux->sd->fd);
333      esp_err_t ret = httpd_ws_respond_server_handshake(&hd->hd_req, uri->supported_subprotocol);
334      if (ret != ESP_OK) {
335          return ret;
336      }
337  }
```

## Reachability

The attacker needs network access to the HTTP or HTTPS port and one request that the parser accepts as a WebSocket upgrade: a GET on a WebSocket URI with Upgrade: websocket, Sec-WebSocket-Version: 13, a Sec-WebSocket-Key, and the malformed Sec-WebSocket-Protocol value. No credentials are involved. The only application code that can run earlier is the optional pre-handshake callback (CONFIG\_HTTPD\_WS\_PRE\_HANDSHAKE\_CB\_SUPPORT), shown above; an application that does not use it, or does not authenticate in it, exposes the path to any client. Server-side TLS does not change this; mutual TLS does, because untrusted clients are rejected during the TLS handshake.

## 03 Impact

The read of address 0 raises a fault that ESP-IDF does not recover from; with the default panic configuration the device reboots. The consequence is a denial of service. There is no write and no information disclosure. Every request repeats it, so a client that keeps sending the request keeps the device from serving.

## Proof

The trigger is the request described under Reachability, with a header value made only of separators, for example:

```
• trigger header (any non-empty value made only of ',' and ' ')
```

**Sec-WebSocket-Protocol: ,**

The defect was found by our libFuzzer campaign on ESP-IDF v5.5.3 and reported to Espressif through its bug bounty programme on 2026-03-17; Espressif fixed it, published the advisory and credited the report. The inputs are kept, but the harness binary kept from that campaign skips this input on purpose, as a known bug, and its source was later rewritten around a different WebSocket parser. The crash was therefore not replayed, and no trace is shown here.

## 04 Fix and recommendation

### Upstream fix

The loop becomes a while (`s != NULL`), so a header that yields no token is treated like an unsupported subprotocol and the handshake completes without a Sec-WebSocket-Protocol response header. The same change replaces the prefix comparison with an exact-length comparison.

```
• 0dc4ee7 · release/v6.0 · components/esp_http_server/src/httpd_ws.c (abridged)

@@ -82,12 +82,15 @@ static bool httpd_ws_get_response_subprotocol(const char *supported_subprotocol,
/* Get first subprotocol from comma separated list */
char *rest = NULL;
char *s = strtok_r(subprotocol, ",", &rest);
- do {
+ if (strncmp(s, supported_subprotocol, strlen(supported_subprotocol)) == 0) {
+ int supported_subprotocol_len = strlen(supported_subprotocol);
+ while (s != NULL) {
+ if (strlen(s) == supported_subprotocol_len &&
+     strncmp(s, supported_subprotocol, supported_subprotocol_len) == 0) {
+     ESP_LOGD(TAG, "Requested subprotocol supported: %s", s);
+     return true;
+ }
+ } while ((s = strtok_r(NULL, ",", &rest)) != NULL);
+ s = strtok_r(NULL, ",", &rest);
+ }

ESP_LOGW(TAG, "Sec-WebSocket-Protocol %s not supported, supported subprotocol is %s", subprotocol,
supported_subprotocol);
```

### 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 37508ab onto the product branch. Where an upgrade is not possible right away, the advisory lists two interim measures: register WebSocket URLs with `supported_subprotocol` left NULL if the application does not need one, or require mutual TLS on the HTTPS server.

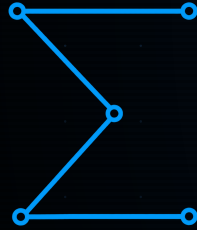
## 05 Timeline

| DATE       | EVENT                                                                   |
|------------|-------------------------------------------------------------------------|
| 2026-03-17 | Reported by Epsilon to the Espressif bug bounty programme               |
| 2026-03-23 | Fix committed to master (9fc0ca1) and release/v6.0 (0dc4ee7)            |
| 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 |
| 2026-05-15 | Advisory GHSA-3j8v-xgrq-5vg8 published                                  |
| 2026-05-19 | ESP-IDF v5.2.7 released with the fix                                    |
| 2026-06-10 | CVE-2026-45541 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 (EunOus), Epsilon. The vendor advisory credits “EunOus / Epsilon Worker”.

- Vendor advisory: <https://github.com/espressif/esp-idf/security/advisories/GHSA-3j8v-xgrq-5vg8>
- CVE record: <https://www.cve.org/CVERecord?id=CVE-2026-45541>
- Fix: [espressif/esp-idf@9fc0ca1](https://github.com/espressif/esp-idf/commit/9fc0ca1) (master), [espressif/esp-idf@0dc4ee7](https://github.com/espressif/esp-idf/commit/0dc4ee7) (release/v6.0)



# ESPILON

SECURITY RESEARCH

Embedded, IoT and OT security research

[espilon.net](https://espilon.net)

[eun0us@espilon.net](mailto:eun0us@espilon.net)

Association loi 1901 • France