Unexpected Pointer Aliasing in IEEE 802154 Fragment Reassembly in Zephyr

Moderate **d3zd3z** published **GHSA-p86r-gc4r-4mq3** on Oct 12, 2021

Package

zephyr (west)

Affected versions

>=2.4.0

Patched versions

v2.5.0

Description

4. Unexpected Pointer Aliasing in IEEE 802154 Fragment Reassembly

- Bug Description: Fragment reassembly cache handling crashes in case a fragment is typed for fragmentation, but itself contains the full payload.
- Bug Result: A NULL pointer is de-referenced as an assumed-to-be-valid pointer to a network packet structure.
- Bug Impact: Denial of Service (DOS) of the target by an attacker sending a single malformed IEEE 802154 fragment.

Bug Details

- Affected code: Fragment reassembly logic in subsys/net/l2/ieee802154/ieee802154_fragment.c#fragment_add_to_cache

High-Level reasoning for bug occurrence:

1. A cache structure holds a list of previously sent ieee802154 which are gathered for later re-assembly
2. Reassembly is only necessary in case multiple (small radio-layer) fragments are required to represent the whole payload.
3. Reassembly logic implicitly assumes that multiple fragments are present before reassembly is triggered
4. This, it is unaware of the situation where the start of the cache list may actually be the newly added fragment itself
5. This leads to an unexpected alias between two pointers, and cleanup logic corrupts the packet structure

Vulnerable code path:

ieee802154_manage_recv_packet->ieee802154_reassemble->fragment_add_to_cache accepts an incoming fragment which is marked for fragmentation

1. If the fragment is the first of its set (indicated by the tag number), a new reassembly cache is requested

- Link:

zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
Line 505 in d969ace

```
505     cache = set_reass_cache(pkt, size, tag);
```

2. It is then checked whether the fragments corresponding to the cache add up to the full indicated payload size

- Link:

zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
Line 517 in d969ace

```
517     if (fragment_cached_pkt_len(cache->pkt) == cache->size) {
```

3. To prepare reassembly, the current packet is assigned the buffer of the first packet in the cache's list

- Link:

zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
Line 519 in d969ace

```
519     pkt->buffer = cache->pkt->buffer;
```

4. To avoid the additional reference in the first fragment in the cache's list, the first list entry's pointer is cleared to NULL

- Link:

zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
Line 520 in d969ace

```
520     cache->pkt->buffer = NULL;
```

BUG: The issue arises where the currently added packet also is the start of the cache list, which arises if the first packet which is sent also contains the full payload. In the following code snippet, this leads to the condition `pkt == cache->pkt`. Thus, the assignment `cache->pkt->buffer = NULL;` sets `pkt->buffer=NULL`.

```
fragment_append(cache->pkt, frag);

if (fragment_cached_pkt_len(cache->pkt) == cache->size) {
        /* Assign buffer back to input packet. */
        pkt->buffer = cache->pkt->buffer;
        cache->pkt->buffer = NULL;

        fragment_reconstruct_packet(pkt);

        ...

        // Assume a valid pkt, and thus pkt->buffer to be properly initialized
}
```

In the following, `net_6lo_uncompress` is called, in which `pkt->buffer` is used and a crash occurs:

```
bool net_6lo_uncompress(struct net_pkt *pkt)
{
        NET_ASSERT(pkt && pkt->frags);

        if ((pkt->frags->data[0] & NET_6LO_DISPATCH_IPHC_MASK) ==
            NET_6LO_DISPATCH_IPHC) {
                        // BUG CRASH TRIGGER: pkt->frags->data[0] crashes, as
                        // pkt->frags==NULL (pkt->frags is an alternative name for pkt->buffer)
                        ...
        }
```

Proposed Fix

- Add check to reassembly logic for the packet which is added also being the first one
- Note that single-frame fragmentation may not be adhering to a specification, so dropping the packet may be an option as well
- Link:

zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
Line 518 in d969ace

```
518      /* Assign buffer back to input packet. */
```

```
@@ -515,9 +528,11 @@ static inline enum net_verdict fragment_add_to_cache(struct net_pkt *pkt)
        fragment_append(cache->pkt, frag);

        if (fragment_cached_pkt_len(cache->pkt) == cache->size) {
-               /* Assign buffer back to input packet. */
-               pkt->buffer = cache->pkt->buffer;
-               cache->pkt->buffer = NULL;
+               if (!first_frag) {
+                       /* Assign buffer back to input packet. */
+                       pkt->buffer = cache->pkt->buffer;
+                       cache->pkt->buffer = NULL;
+               }

                fragment_reconstruct_packet(pkt);
```

Patches

This has been fixed in:

- main: Fixed here
- v1.14: not-fixed

For more information

If you have any questions or comments about this advisory:

- Open an issue in zephyr
- Email us at Zephyr-vulnerabilities

embargo: 2021-04-21
zepsec: ZEPSEC-115

Severity

Moderate 6.5 / 10

CVSS base metrics

Attack vector	Adjacent
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchanged
Confidentiality	None
Integrity	None
Availability	High

CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVE ID

CVE-2021-3322

Weaknesses

CWE-476