Integer Underflow in 6LoWPAN IPHC Header Uncompression in Zephyr

High d3zd3z published GHSA-89j6-qpxf-pfpc on Oct 12, 2021

Package
zephyr (west)

Affected versions	Patched versions
>=2.4.0	2.5.0

Description

5. Integer Underflow in 6LoWPAN IPHC Header Uncompression

- Bug Description: Missing checks on network packet size in uncompress_IPHC_header leads to an integer underflow, resulting in corrupted net_buf bounds
- Bug Result: The size field of a net_buf_simple struct gets underflown, significantly enlarging the assumed size of a network buffer, leading to out-of-bounds accesses in IPv6 functionality.
- Bug Impact: Out-of-bounds accesses inside IPv6 parsing logic. I highly suspect this to be exploitable to Arbitrary Code Execution by an attacker sending IPv6 packets (such as maliciously fragmented ICMPv6 Ping requests).

Bug Details

- Affected code: Header Uncompression logic in subsys/net/ip/6lo.c#net_6lo_uncompress->uncompress_IPHC_header.

High-Level reasoning for bug occurrence:

1. A recent fix of a bug which I reported earlier added a check to make sure enough buffer tail space was available and that the right amount of bytes were moved during in-place 6LoWPAN IPHC header compression(

   zephyr/subsys/net/ip/6lo.c
   Line 1356 in d969ace

   ```
   memmove(cursor, frag->data, frag->len - diff);
   ```

)
2. Fuzzing has shown another bug in the handling of the opposite case where the header cannot be uncompressed into the existing buffer.
3. If not enough space is available in the current buffer, an additional buffer is allocated to hold the uncompressed contents
4. In this external-buffer uncompressed handling, the size of the compressed header is calculated based on the metadata of the initial parts of the payload
5. The expected header of the pre-computed size is then stripped from the original fragment using net_buf_pull.
6. It is not checked, however, if the buffer held enough bytes to contain this uncompressed header in the first place
7. As a result, net_buf_pull will try to remove more bytes from the buffer than are actually present, which makes the buffer's size go negative, which is a large number as it is later interpreted as an unsigned number.
8. This sets the buffer pointer of the net_buf_simple buffer out of bounds, such that out-of-bounds memory is treated as a way too large network buffer

zephyr/subsys/net/ip/6lo.c
Line 1366 in d969ace

```
net_buf_pull(pkt->buffer, compressed_hdr_size);
```

Vulnerable code path:

0. Multiple code paths lead to this IP-layer uncompression logic, including Bluetooth, CAN bus, and IEEE 802.15.4 radio packets. I list a sample code path from ieee802154 packets, which were fuzzed using our research prototype.
1. ieee802154_recv->ieee802154_manage_recv_packet->ieee802154_reassemble->fragment_add_to_cache
   - Fragments are added via fragment_add_to_cache, and the full packet eventually reconstructed if all required fragments are present
   - Link:

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

     ```
     if (fragment_cached_pkt_len(cache->pkt) == cache->size) {
     ```
2. fragment_add_to_cache->net_6lo_uncompress
   - After reconstructing the packet data, the compressed IPv6 header needs to be uncompressed
   - Link:

     zephyr/subsys/net/l2/ieee802154/ieee802154_fragment.c
     Line 527 in d969ace

     ```
     if (!net_6lo_uncompress(pkt)) {
     ```
3. fragment_add_to_cache->net_6lo_uncompress->uncompress_IPHC_header->get_ihpc_inlined_size
   - To uncompress the header, first the length is determined from the header's 'iphc' tag
   - Link:

     zephyr/subsys/net/ip/6lo.c
     Line 1330 in d969ace

     ```
     inline_size = get_ihpc_inlined_size(iphc);
     ```
4. fragment_add_to_cache->net_6lo_uncompress->uncompress_IPHC_header
   - Without checking the actual size of the incoming packet's buffer, the buffer is trimmed by the size which was computed from the 'iphc' tag
   - Link:

     zephyr/subsys/net/ip/6lo.c
     Line 1366 in d969ace

     ```
     net_buf_pull(pkt->buffer, compressed_hdr_size);
     ```
5. uncompress_IPHC_header->net_buf_simple_pull

- The network buffer API function net_buf_pull will then substract the size, underflowing it in the process
- Link:

zephyr/subsys/net/buf.c
Line 1126 in d969ace

```
1126      buf->len -= len;
```

Proposed Fix

- After calculating the size of the expected uncompressed header based on the iphc metadata field, check that enough space is actually present within the buffer.
- Note that the single pkt->buffer fragment may not represent all data within the packet, which may consist of multiple fragments
  - Legitimate parties will probably not send a too small fragment as part of the fragments, so the first fragment not holding the full compressed header could be used as an indication to just drop the packet
  - Otherwise, the logic would have to support stripping the uncompressed header from across multiple network packet fragments/buffers
- If the uncompressed header is expected to be present in the first fragment, the following check could be implemented:

```
diff --git a/subsys/net/ip/6lo.c b/subsys/net/ip/6lo.c
index 736cf05839..f870abf4fc 100644
--- a/subsys/net/ip/6lo.c
+++ b/subsys/net/ip/6lo.c
@@ -1348,6 +1348,12 @@ static bool uncompress_IPHC_header(struct net_pkt *pkt)
                        nhc_inline_size;
        }

+       /* Proposed fix: Make sure the buffer holds the full compressed header */
+       if (compressed_hdr_size > pkt->buffer->len) {
+               NET_ERR("Too small packet to hold compressed IPHC header");
+               return false;
+       }
+
        if (net_buf_tailroom(pkt->buffer) >= diff) {
                NET_DBG("Enough tailroom. Uncompress inplace");
                frag = pkt->buffer;
```

Patches

This has been fixed in:

- main: #31971
- v1.14: NA

For more information

If you have any questions or comments about this advisory:

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

embargo: 2021-04-14
zepsec: ZEPSEC-116

Severity

High 8.3 / 10

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Changed
Confidentiality	Low
Integrity	Low
Availability	Low

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

CVE ID

CVE-2021-3323

Weaknesses

CWE-191