Potential Vulnerabilities in RIOT

Vuln 1: Out-of-Bound Read Vulnerability in 6LoWPAN

Brief Description

In _iphc_ipv6_decode, the payload_offset variable is continuously incremented and used to access the iphc_hdr, but there is no validation that it is always within the bounds of the received packet. As a result, if the received packet is not large enough to contain the expected length of the compressed IPv6 header, this will result in reading out of bounds when the IPHC header is decoded. The _iphc_ipv6_decode function is called from gnrc_sixlowpan_iphc_recv, which processes received IPHC compressed packets. Hence, this vulnerability can be triggered by an attacker.

Technical Details

- The IPHC header is read from the sixlowpan payload in line 736.
- There is only a validation that the size is not smaller than 2 bytes in line 744.
- The received data is used to call the _iphc_ipv6_decode function in line 776.
- The payload_offset is initialized to 2 in line 173 and continuously incremented throughout the _iphc_ipv6_decode function without any validation.
- This payload_offset variable is also used to access the iphc_hdr field in multiple locations, including as the source address in a memcpy field in line 255.

Vuln 2: Out-of-Bound Write and Read in 6LoWPAN

Brief Description

In the _iphc_nhc_ipv6_ext_encode function, the ext_len field, which is read from the packet in line 1562, is used to read and copy data into the nhc_data pointer (line 1450) without validation.

Also, there is a potential OOB read vulnerability in the same line 1450 as well. The ext_len field is validated using an assert statement in line 1565 that it is within the bounds of the received packet. However, note that in a non-debug build, this assertion can be a no-op. As discussed in RIOT's CVE-2024-32018, assertions are not a reliable defense against untrusted input as most codebases define assertion macros as no-op in non-debug builds.

Possible Vuln 3: Potential Null Pointer Dereferencing in 6LoWPAN

Brief Description

In gnrc_sixlowpan_iphc_recv, the interface returned by the gnrc_netif_hdr_get_netif function in line 775 is not validated to be non-null before being used. While the gnrc_netif_hdr_get_netif function can potentially return NULL (line 486 in gnrc_netif.c), we are not sure if this is a realistic scenario in practice (maybe, this is a possible scenario if an interface can be freed while still being referenced).

Additionally, we also note that validations on the pointer returned by the gnrc_netif_hdr_get_netif function are inconsistent. It is validated in a couple of locations (e.g. here and here) but not validated in others.

Possible Vuln 4: Arithmetic Underflow in URI Parser

Brief Description

In the _consume_authority function of uri_parser.c, the value returned by the _strchrb function is not correctly validated. While _strchrb can return NULL if the searched character (‘]’) does not exist (line 41), the _consume_authority function only validates that the returned value is less than the authority_end field (which NULL satisfies) (line 177). This will further lead to an integer underflow when the result->ipv6addr_len field is computed (line 195), and can potentially cause an out-of-bound read or write when used without validation.

We have not yet located a concrete flow where this bug leads to reading or writing out of memory. However, since this is a utility routine used in many other subsystems, protocols and applications, we suggest a patch here might be helpful.

Possible Vuln 5: Out-of-Bound Read in 6LoWPAN

Brief Description

In the _iphc_ipv6_encode function, the ipv6 header is read from the pkt->next->data field (line 1092). However, there is no validation that the data in this next header is not NULL or contains up to the size of the IPv6 header. This field is repeatedly accessed throughout this function.

The _iphc_ipv6_encode function is reachable from the _encode_frag_for_forwarding function which forwards received packets. We see that the parent function gnrc_sixlowpan_iphc_recv validates the size of the received packet (line 755). However, we are unable to verify if the

pkt->next pointer in _iphc_ipv6_encode always points to this validated ipv6 pointer in the gnrc_sixlowpan_iphc_recv function.