Potential Vulnerabilities in RIOT

Vuln 1: Out-of-Bound Write in Bluetooth Subsystem

Description

There is a potential OOB write in the gen_prov_cont function of the pb_adv.c file. This vulnerability occurs because the seg variable, which is user-controlled, is used to compute the offset where data is copied into. However, an integer underflow and overflow invalidates the validation on seg in line 529.

Technical Details

1. seg is computed from rx->gpc in line 483.
2. There is a validation in line 529 that the offset computed from seg ((20 + ((seg - 1) * 23))) and the length of data does not overflow the receiver buffer.
3. However, if seg = 0, the computation will yield an offset of 253
    a. seg - 1 underflows, yielding 255
    b. 255 * 23 overflows multiple times, yielding 233
    c. 233 + 20 yields 253
4. If buf->len has a value greater than 2, then adding it to 253 overflows, yielding a number that is potentially less than RX_BUFFER_MAX (65).
5. Finally, the computed offset is used to compute the destination address of the memcpy in line 534. This can cause a large out-of-bound write in the rx.buf->data array.

Potential Impact

An out-of-bound write can lead to an arbitrary code execution. This is more severe in real-time operating systems like Zephyr that run in embedded devices without common memory protection systems. Even on devices with some form of memory protection, this can still lead to a crash and a resultant denial of service.

Recommended Fix

Based on our analysis, we found that this vulnerability only occurs if seg has a value of 0. Hence, we suggest adding a validation that seg is not 0.

Vuln 2: Out-of-Bound Write in Bluetooth Subsystem

Description

There is a potential OOB Write vulnerability in the gen_prov_start function in pb_adv.c. The full length of the received data is copied into the link.rx.buf receiver buffer in line 659 without any validation on the data size.

Technical Details

1. There is a memcpy on line 659 that copies data into link.rx.buf->data buffer.
2. There is no validation on the length of the received data buf->len.
3. There seems to be a check on line 617. However, link.rx.buf->len is read from the packet (in line 604) and may not correspond to the actual length of the packet in buf->len.
4. Hence, if the source buffer has a length greater than that of the receiver buffer, a malicious packet can write out of bounds in this function.

Potential Impact

An out-of-bound write can lead to an arbitrary code execution. This is more severe in real-time operating systems like Zephyr that run in embedded devices without common memory protection systems. Even on devices with some form of memory protection, this can still lead to a crash and a resultant denial of service.

Recommended Fix

We recommend that the validation on line 617 should be updated so it validates that the buf->len field is not greater than the receiver buffer size.