Potential Vulnerabilities in RIOT-OS

Vuln 1: Out-of-Bound Write in _gcoap_forward_proxy_copy_options

Description

There is a potential out-of-bound write vulnerability in line 411 of the _gcoap_forward_proxy_copy_options function. This occurs because the payload is copied from client_pkt->payload to pkt->payload but there is no check that the size of pkt->payload is up to or greater than the size of data copied in.

Technical Details

- The function _gcoap_forward_proxy_copy_options performs a memcpy operation, copying data from client_pkt to pkt, and is called from the _gcoap_forward_proxy_via_coap function.
- The destination packet, pkt, is initialized in line 459, and looking at the implementation of coap_pkt_init, the payload field will have a size equal to (CONFIG_GCOAP_PDU_BUF_SIZE - sizeof(coap_hdr_t) - token_len).
- CONFIG_GCOAP_PDU_BUF_SIZE by default, has size 128 bytes. Hence, the payload length of the destination packet has a max size of 124 bytes (assuming 0 token_len and coap_hdr_t of 4 bytes.
- Tracing the source of the client pkt up to _forward_proxy_handler, we can't find any constraints on the size of the payload that client_pkt holds.
- Hence, if client_pkt holds a user-provided data or data of arbitrary length, then this will cause a buffer overflow in line 411.

Impact

An out-of-bound write can lead to an arbitrary code execution. This is more severe in real-time operating systems like RIOT 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.

Fix Recommendation

We recommend adding a validation before line 411 to ensure pkt->payload has enough space for the incoming data. The function can return an error if the validation fails. Here is a PR that makes this change.

Vuln 2: Out-of-Bound Read in _gcoap_forward_proxy_copy_options

Description

The function _gcoap_forward_proxy_copy_options also has an out-of-bound read in line 585 when the _cep_set_req_etagis called from line 381. This occurs because _gcoap_forward_proxy_copy_options does not validate optlen, which is user-controlled, before using it to copy data from value in line 585.

Technical Details

- This vulnerability occurs because the variable, optlen, is user-controlled and not validated.
- They are read from the packet using the coap_opt_get_next function call in line 366.
- Reviewing the call trace, we see that coap_opt_get_next leads to _decode_value getting called in line 243.
- _decode_value returns a value read from the user-controlled packet, after undergoing some computation.
- This value is saved in the opt_len field without any validation.
- Finally, this unvalidated optlen value is used to perform a memcpy read operation in line 585.

Impact

An out-of-bound memory access can cause a crash and lead to a denial of service. Such denial of service can have severe consequences in real-time safety-critical devices where RIOT-OS is typically used.

Fix Recommendation

We recommend adding a validation that the optlen field, read from line 366, is within the bounds of the client_pkt buffer. Here is a PR that makes this change.

Vuln 3: Out-of-Bound Write in _rbuf_add

Description

There is a potential out-of-bound write in the memcpy operation on line 493 of the _rbuf_add function in gnrc_sixlowpan_frag_rb.c.
This occurs because an arithmetic overflow in line 388 can cause an invalid offset parameter to be used to compute the destination address in line 493.

Technical Details

- Line 388 validates that the offset and frag_size would not overflow the buffer in entry.rbuf->pkt->data.
- However, if the offset is very large, the addition will overflow, leading to a value less than entry.super->datagram_size. As a result, the validation passes.
- The rbuf_add function can be reached from the _handle_nth_rfrag function in the gnrc_sixlowpan_frag_sfr.c file (line 1224).
- In this trace, the offset is retrieved using the sixlowpan_sfr_rfrag_get_offset function, which only reads the offset field from the packet. Hence, the offset field is user-controlled.
- There is a call to a valid offset check in line 311 in the _rbuf_add function. However, for SFR fragments, this check only verifies that the offset is not equal to zero. Hence, it is not sufficient to prevent this vulnerability.

Impact

An out-of-bound write can lead to an arbitrary code execution. This is more severe in real-time operating systems like RIOT 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.

Fix Recommendation

We recommend that the offset validation on line 388 includes a check that the offset field is within the range of entry.super->datagram_size or that the sum (offset+frag_size) does not overflow. Here is a PR that makes this change.

Vuln 4: Out-of-Bound Read in _rbuf_add

Description

There is an out-of-bound read in the memcpy operation on line 493 in the _rbuf_add function. This occurs because there is a discrepancy in how the first and the nth fragments are distinguished in the _6lo_frag_payload and the _6lo_frag_size functions. This vulnerability can be triggered by a malicious nth fragment with an offset value of 0.

Technical Details

- For a non-SFR fragment, the data and frag_size fields are calculated using the _6lo_frag_payload and the _6lo_frag_size functions respectively.
- The _6lo_frag_payload function uses the sixlowpan_frag_1_is function to distinguish a first from an nth fragment. On the other hand, the _6lo_frag_size function only checks if the offset is 0 or not.
- Hence, for an nth packet with an offset field of 0, the frag_size is computed as if it was a first fragment, leading to an off-by-one error.
- Since frag_size is used to copy out data from the data field in line 493, this causes an OOB read of 1 byte.
- Reviewing the callers of _rbuf_add, we find that the offset field for an nth fragment is read from the packet, hence, user controlled. This means it can have a 0 value.
- Additionally, the _valid_offset function called in line 311 only checks that offset variable corresponds with what was read in the packet (here).
- Hence, the 6Lowpan stack does not properly validate that nth fragments have valid offsets, leading to an invalid memory access.

Impact

An out-of-bound memory access can cause a crash and lead to a denial of service. Such denial of service can have severe consequences in real-time safety-critical devices where RIOT-OS is typically used.

Fix Recommendation

We recommend modifying the _6lo_frag_size function so that it uses the sixlowpan_frag_1_is function to distinguish first from nth fragment. Here is a PR that makes this change.

Vuln 5: Out-of-Bound Read and Null Pointer Dereferencing in _get_content_format

Description

The function _get_content_format reads the content_type from a received packet. However, if the content_type option is at the end of the packet, this function, in the memcpy on line 330, can read up to two bytes beyond the end of the packet. Additionally, _parse_option can return NULL and pkt_pos is dereferenced in lines 328 and 330 without any null pointer validation.

Technical Details

- The _get_content_format function can be reached from multiple functions, such as the _on_rd_init function in cord_lc.c.
- The option_len field is read using the _parse_option function in line 323.
- However, there is no validation that the complete option_len is within the bounds of the packet.
- Hence, if option_len is 2 and pkt_pos is already at the end of the packet, the memcpy read operation in line 330 will lead to an out-of-bound memory access.
- Additionally, we see that _parse_option can also return null if the current pointer position is already invalid (line 237).
- However, even if this happen, the _get_content_format function goes ahead to dereference or read from the null pointer in lines 328 and 330 respectively.

Impact

An out-of-bound memory access or null pointer dereferencing can cause a crash and lead to a denial of service. Such denial of service can have severe consequences in real-time safety-critical devices where RIOT-OS is typically used. A skilled attacker can also use them to craft more severe exploits.

Fix Recommendation

We recommend adding a validation after the _parse_option call in line 323 that the returned pkt_pos is not NULL and the option_len field read can be safely accessed. Here is a PR that makes this change.