Potential Vulnerabilities in Zephyr

Out of bounds read when calling crc16_ansi and strlen in dns_validate_msg

Description

The function dns_validate_msg in subsys/net/lib/dns/resolve.c validates incoming DNS messages. However, due to an incorrect validation, a malicious or malformed DNS packet without a payload can cause an out-of-bounds read, resulting in a crash (denial of service) or an incorrect computation.

Technical Details

The target function (dns_validate_msg) validates and processes received DNS packets. However, for packets with DNS ID of 0, QD count of 1 and a missing payload, the crc16_ansi and strlen functions in lines 857-858 will read out-of-bound.

Here is the program flow.

- In line 677, *dns_id may be read as 0.
- In line 698, qdcount is read from the head as 1. This check passes
- dns_unpack_response_query is called in line 706.
    - query_offset is set to value 12 in line 327.
    - Remaining_size is computed as 0 in line 329.
    - rc is 0 in line 331. The dns_unpack_response_query exits in line 333 with a negative value.
- As *dns_id has value 0, the dns_validate_msg function does not quit in line 711, despite the returned error. The function progresses as usual.
- As ancount is 0, this loop in line 731 is not executed.
- In line 852, *query_idx is still -1. Hence, this if block is executed.
- In line 856, query_name pointer is calculated to point to the 13th byte. As the packet only has 12 bytes, this pointer now points out-of-bounds.
- In lines 857 and 858, crc16_ansi and strlen is called with the invalid query_name pointer. This leads to an out-of-bounds read of multiple bytes.

Here is a sample packet that can cause this.
{0, 0, 128, 1, 0, 1, 0, 0, 0, 0, 0, 0};

Impact

In devices with memory protection, this out-of-bound reads will lead to a crash, causing denial of service. In safety-critical devices, this can have severe consequences.
In embedded devices without memory protection, this can cause an invalid computation that impacts device behavior.

Fix Recommendation

We recommend adding a DNS payload validation that verifies that the qdcount and ancount values present in the header are correct.

Out of bounds read when unpacking DNS answers

Description

The dns_unpack_answer function in dns_pack.c decodes DNS answers from incoming DNS data. A lack of input validation allows for out of bounds reads caused by malicious or malformed packets.

These functions are mostly called by dns_unpack_answer (line 109 in subsys/net/lib/dns/dns_pack.c as of 6798064), which is responsible for decoding incoming DNS data. There are many cases where these out of bounds failures occur, but this report will be focusing on a specific case and a general solution that will correct all out of bounds reads.

Technical Details

The target function is called when validating received DNS messages. Here is a vulnerable program flow.

- The answer pointer in line 118 is computed to point to the start of the DNS answers.
- dname_len is calculated in line 120.
- rem_size is computed and validated in line 137. However, this validation is wrong as it does not recognize the answer_offset. rem_size is computed with respect to the start of the packet, instead of the start of the answer region.
- Hence, all accesses to the buffer pointed to by answer (lines 152 to 169) can potentially lead to an out-of-bound read.

This vulnerability can be exposed with the following packet.
```
uint8_t msg[18] = {7, 7, 141, 128, 0, 1, 0, 1, 25, 158, 96, 70, 0, 0, 1, 0, 1, 0};
```

Impact

This out-of-bounds read can cause a crash and lead to a denial of service.

Fix Recommendation

This vulnerability can be fixed by updating the computation of rem_size in line 137.

```
rem_size = dns_msg->msg_size - dns_msg->answer_offset - dname_len;
if (rem_size < 2 + 2 + 4 + 2) {
                return -EINVAL;
        }
```

Out of bounds read in dns_copy_qname

Description

The function dns_copy_qname in dns_pack.c performs performs a memcpy operation with an untrusted field and does not check if the source buffer is large enough to contain the copied data.

Technical Details

The dns_copy_qname function contains a memcpy operation in line 403. The lb_size argument is read from the packet in line 377.
The function only validates the destination buffer in line 397 to ensure the buffer is large enough to contain the expected data. However, there is no validation for the source buffer.

Impact

For embedded devices with memory protection, this out-of-bound read can cause a crash and a denial of service. For devices without memory protection, this can lead to incorrect and unexpected behaviors.

Fix Recommendation

In the target function, we can correct the size check condition (line 397) to prevent these reads from occurring. We can change the condition in line 397 to:

```
DNS_LABEL_LEN_SIZE + lb_size > MIN(size - *len, msg_size - pos)
```