Out-of-bound write vulnerability in the Bluetooth mesh core stack can be triggered during provisioning

High **ceolin** published **GHSA-p449-9hv9-pj38** on Jul 25

Package

zephyr (west)

Affected versions

<= 3.0

Patched versions

None

Description

Impact

In Zephyr bluetooth mesh core stack, an out-of-bound write vulnerability can be triggered during provisioning, because there lacks a check for mismatched SegN and TotalLength in Transaction Start PDU.

In gen_prov_start, there lacks a check for mismatched SegN and TotalLength. For example, TotalLength 65 with SegN 62 in Transaction Start PDU is considered as valid (infact, if TotalLength is 65, SegN should be only 2). SegN 62 will be set into `link.rx.last_seg` .

```
BT_DBG("len %u last_seg %u total_len %u fcs 0x%02x", buf->len,
       START_LAST_SEG(rx->gpc), link.rx.buf->len, link.rx.fcs);

if (link.rx.buf->len < 1) {
        BT_ERR("Ignoring zero-length provisioning PDU");
        prov_failed(PROV_ERR_NVAL_FMT);
        return;
}

if (link.rx.buf->len > link.rx.buf->size) {
        BT_ERR("Too large provisioning PDU (%u bytes)",
               link.rx.buf->len);
        prov_failed(PROV_ERR_NVAL_FMT);
        return;
}

if (START_LAST_SEG(rx->gpc) > 0 && link.rx.buf->len <= 20U) {
        BT_ERR("Too small total length for multi-segment PDU");
        prov_failed(PROV_ERR_NVAL_FMT);
        return;
}

prov_clear_tx();

link.rx.last_seg = START_LAST_SEG(rx->gpc);
```

By sending malformed Transaction Start PDU with legal TotalLength and oversize SegN, the check for SegO and SegN in Transaction Continue PDU can be bypassed.

```
if (seg > link.rx.last_seg) {
        BT_ERR("Invalid segment index %u", seg);
        prov_failed(PROV_ERR_NVAL_FMT);
        return;
}
```

In consequence, sending a Transaction Continue PDU with actually oversized (i.e., larger than 2, corresponding to the size of rx_buf) SegO will trigger out-of-bound write. That is, if SegO > 2, then 20 + (SegO - 1)×23 + 23 > 65,

```
NET_BUF_SIMPLE_DEFINE_STATIC(rx_buf, 65);

memcpy(XACT_SEG_DATA(seg), buf->data, buf->len);
XACT_SEG_RECV(seg);
```

where 20 + (SegO - 1)×23 is the offset.

Patches

This has been fixed in:

main: #45136
v3.0: #45188
v2.7: #45187

Credits

Han Yan(闫晗),Lewei Qu(曲乐炜),Dongxiang Ke(柯懂湘) of Baidu AIoT Security Team

For more information

If you have any questions or comments about this advisory:

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

embargo: 2022-06-19

Severity

High 8.2 / 10

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

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

CVE ID

CVE-2022-1041

Weaknesses

CWE-787