New issue

Jump to bottom

Potential buffer out-of-bound read in gatt-database.c:cli_feat_read_cb #70

Closed **zxtwonder** opened this issue on Jan 4, 2021 · 7 comments

zxtwonder commented on Jan 4, 2021 • edited

bluez/src/gatt-database.c
Lines 1078 to 1079 in 35a2c50

```
1078      len = sizeof(state->cli_feat) - offset;
1079      value = len ? &state->cli_feat[offset] : NULL;
```

```
static void cli_feat_read_cb(struct gatt_db_attribute *attrib,
                                        unsigned int id, uint16_t offset,
                                        uint8_t opcode, struct bt_att *att,
                                        void *user_data)
{
...
        size_t len = 0;
...
        len = sizeof(state->cli_feat) - offset;
        value = len ? &state->cli_feat[offset] : NULL;
```

Both `len` and `sizeof()` are unsigned and `offset` is an external input without validation. It seems like a malicious GATT client can cause out-of-bound read on memory locations after the given device_state::cli_feat pointer.

Similar validation like below should be sufficient:

bluez/src/gatt-database.c
Line 691 in 35a2c50

```
691      if (offset > 2) {
```

github-actions bot pushed a commit to BluezTestBot/bluez that referenced this issue on Jan 4, 2021

gatt: Fix potential buffer out-of-bound ... 9e6889d

BluezTestBot mentioned this issue on Jan 4, 2021

[PW_SID:408793] [BlueZ] gatt: Fix potential buffer out-of-bound BluezTestBot/bluez#640

Closed

github-actions bot pushed a commit to tedd-an/bluez that referenced this issue on Jan 4, 2021

gatt: Fix potential buffer out-of-bound ... 61ed28e

tedd-an mentioned this issue on Jan 4, 2021

[PW_SID:408793] [BlueZ] gatt: Fix potential buffer out-of-bound tedd-an/bluez#839

Closed

zxtwonder commented on Jan 4, 2021 Author

@Vudentz Thanks for the super quick update. One question on the change though - I only have a practical understanding on the BT spec, but should we only return `BT_ATT_ERROR_INVALID_OFFSET` when `offset` is strictly larger than `sizeof(state->cli_feat)` - to give the server a chance to indicate the end of the data?

```
        if (offset >= sizeof(state->cli_feat)) {
                ecode = BT_ATT_ERROR_INVALID_OFFSET;
                goto done;
        }
```

Vudentz commented on Jan 4, 2021 Contributor

> **@Vudentz** Thanks for the super quick update. One question on the change though - I only have a practical understanding on the BT spec, but should we only return `BT_ATT_ERROR_INVALID_OFFSET` when `offset` is strictly larger than `sizeof(state->cli_feat)` - to give the server a chance to indicate the end of the data?
>
> ```
>         if (offset >= sizeof(state->cli_feat)) {
>                 ecode = BT_ATT_ERROR_INVALID_OFFSET;
>                 goto done;
>         }
> ```

If the offset is == sizeof(state->cli_feat) it would be already past the end as the offset start at index 0 so valid offsets are in a range of 0 to size of attribute - 1.

zxtwonder commented on Jan 4, 2021 Author

Based on the previous code, I was under the impression that the client needs a return of 0 length to be sure the read is complete.

```
        len = sizeof(state->cli_feat) - offset;
        value = len ? &state->cli_feat[offset] : NULL;
```

But after a further look, it seems that gatt-database does not handle offset consistently.

Sometimes, it allows for empty read:

```
static void gap_appearance_read_cb(struct gatt_db_attribute *attrib,
...
        uint8_t appearance[2];
...
        if (offset > 2) {
```

But other times not:

```
static void gatt_ccc_read_cb(struct gatt_db_attribute *attrib,
...
        const uint8_t *value = NULL;
...
        if (offset) {
```

I guess it is all corner case now since these are all short values. I will have to read the spec to see what the client should expect then.

tedd-an closed this as completed in `tedd-an/test-bluez@3a40bef` on Jan 5, 2021

---

setharnold commented on Jun 8, 2021

Hello, please use CVE-2021-3588 for this issue. Thanks.

jefferyto commented on May 17

This issue appears to have been closed by a test commit and was never actually fixed.

Vudentz commented on May 17 — Contributor

@jefferyto it has been pushed: https://git.kernel.org/pub/scm/bluetooth/bluez.git/commit/?id=3a40bef49305f8327635b81ac8be52a3ca063d5a

jefferyto commented on May 18

Apologies for the noise (the fix was also replaced in `6a50b6a`).

Haotian-Shi-cyber mentioned this issue on Jun 1

[version 5.64]failed headphone connection #351

Closed

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

[PW_SID:408793] [BlueZ] gatt: Fix potential buffer out-of-bound
BluezTestBot/bluez

[PW_SID:408793] [BlueZ] gatt: Fix potential buffer out-of-bound
tedd-an/bluez

4 participants