Practical Binary Analysis

Errata and Corrections

Updated 2022-01-06 07:56:17+01:00

If you find any errors not listed here, please report them to No Starch Press or to the author at da.andriesse@gmail.com. Errors listed in blue are (currently being) fixed in newer ebook releases and reprints. If you bought the book at No Starch Press, you should be able to download the updated ebook version from your account.

Page xiii (ToC) Some PDF viewers show the title "Binary Instruction" instead of "Binary Instrumentation" for Ch9 in the table of contents.

Page xviii (Foreword) "programs programs" should read "programs."

Page 2 (Introduction) "this books" should read "this book."

Page 6 (Introduction) The ePub and Mobi versions of the book show the assembly example in AT&T syntax. It should be in Intel syntax. The PDF version does not have this error.

Page 55 (Ch2) In exercise 3, "Use `readelf` to disassemble two binaries" should read "use `objdump` to disassemble two binaries."

Page 110 (Ch5) In Listing 5-10, the command reads "`objdump -d ctf`." This should read "`objdump -M intel -d ctf`." (Without the option `-M intel` the command will output assembly in AT&T syntax instead of Intel syntax.)

Page 136 (Ch6) The URL mentioned in footnote 10 is no longer available. An alternative URL is `https://www.cs.vu.nl/~herbertb/papers/howard_ndss11.pdf`.

Page 144 (Ch6) "... make **flow-sensitive** analysis too computationally expensive" should read "... make **context-sensitive** analysis too computationally expensive."

Page 153 (Ch6) "You'll probably need to use some inline **disassembly** to achieve this" should read "You'll probably need to use some inline **assembly** to achieve this."

Page 168 (Ch7) The text states that "paths given in `LD_PRELOAD` need to be absolute." While this is indeed a best practice to account for (sub)processes that may change the working directory, it is not a strict requirement. Relative paths will work in processes that leave the working directory unchanged.

Page 199 (Ch8) The ePub and Mobi versions of the book contain an error in Listing 8-4, near marker ❹. The comparison `n >= 0` should read `n <= 0`. The PDF version does not have this error.

Page 207 (Ch8) At the top of the page, the line

```
if(target && !seen[target] && text->contains(target))
```

causes the recursive disassembler to ignore branches to address zero. This is fine on modern Linux distributions and most platforms, but could be an issue on esoteric platforms that allow mapping code at address zero.

Page 217 (Ch8) The code near marker ❹ contains two integer underflows:

- The `for` loop's initial value `uint64_t a = root-1` will underflow if `root` equals zero (this can only happen if the `.text` section is mapped at VMA zero).
- If `root` contains a value less than `root_offset`, the loop condition `a >= root-root_offset` will underflow. For example, if `root` equals zero, `root-root_offset` will wrap around and assume the value `UINT64_MAX - root_offset + 1`.

The underflows can be fixed by checking before the loop that `root > 0 && root >= root_offset` and changing the loop header to read:

```
for(uint64_t a = root-1;
    text->contains(a) && a >= root-root_offset;
    a--)
```

Page 386 (Appendix A) "After setting up a basic function frame, main decrements `rsp` by `0x10` bytes to reserve room for two 8-byte local variables on the stack" should read "After setting up a basic function frame, main decrements `rsp` by `0x10` bytes to reserve room for local variables on the stack (four bytes for `argc` and eight bytes for the `argv` pointer, the remaining bytes being used for padding)."

Page 405 (Appendix B) "The `write_shdr` function takes three parameters" should read "The `write_shdr` function takes four parameters."