

Kristof Beyls  
EuroLLVM 2022

# LLSOFTSECBOOK: AN OPEN SOURCE BOOK ON LOW-LEVEL SOFTWARE SECURITY FOR COMPILER DEVELOPERS

# WHY?

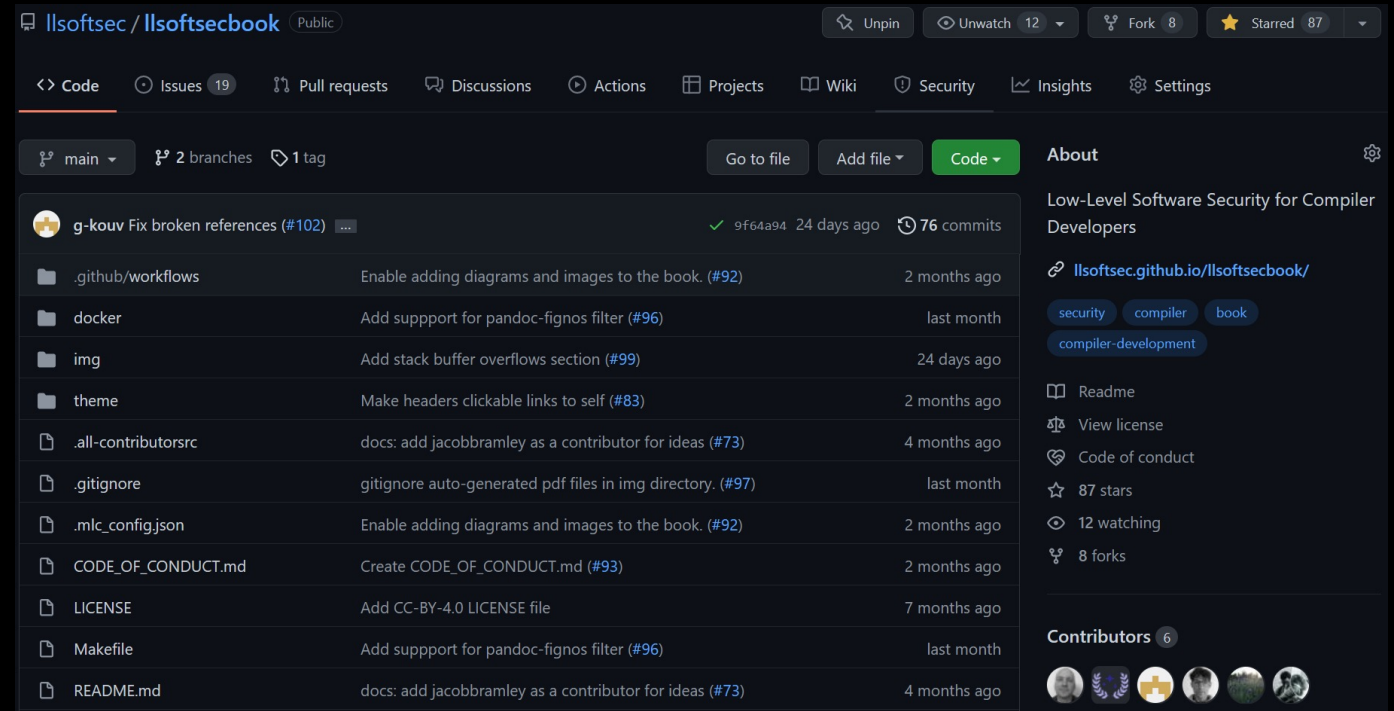
- Many compiler engineers working on security feel they lack deep enough expertise.
  - Did I really make life more difficult for attackers?
  - Could this bug be an exploitable security issue?
- We did not find good educational material
  - Lots of detailed exploits are described
  - But little material on “this is what compiler engineers should know about security”

# IS IT WORTH CREATING A BOOK?

- No better way to learn about a topic then trying to describe it (Feynman technique)
- ✓ Compiler engineers are increasingly working on security features.
- ✓✓ A book helps all compiler engineers, not just me.
- ✓✓✓ An open source book enables:
  - Contributions from experts where we lack expertise.
  - Might help to strengthen the network of low-level software security experts

# HOW?

- Licensed under CC-BY-4.0 license
- <https://github.com/llsoftsec/llsoftsecbook>



The screenshot shows the GitHub repository page for `llsoftsec / llsoftsecbook`. The repository is public and has 12 unwatchers, 8 forks, and 87 stars. The main navigation bar includes links for Code, Issues (19), Pull requests, Discussions, Actions, Projects, Wiki, Security, Insights, and Settings. The repository is currently on the `main` branch, with 2 other branches and 1 tag. The file list shows the following structure:

File/Folder	Description	Last Commit
<code>.github/workflows</code>	Enable adding diagrams and images to the book. (#92)	2 months ago
<code>docker</code>	Add support for pandoc-fignos filter (#96)	last month
<code>img</code>	Add stack buffer overflows section (#99)	24 days ago
<code>theme</code>	Make headers clickable links to self (#83)	2 months ago
<code>.all-contributorsrc</code>	docs: add jacobbramley as a contributor for ideas (#73)	4 months ago
<code>.gitignore</code>	gitignore auto-generated pdf files in img directory. (#97)	last month
<code>.mlc_config.json</code>	Enable adding diagrams and images to the book. (#92)	2 months ago
<code>CODE_OF_CONDUCT.md</code>	Create CODE_OF_CONDUCT.md (#93)	2 months ago
<code>LICENSE</code>	Add CC-BY-4.0 LICENSE file	7 months ago
<code>Makefile</code>	Add support for pandoc-fignos filter (#96)	last month
<code>README.md</code>	docs: add jacobbramley as a contributor for ideas (#73)	4 months ago

The right sidebar contains the repository's description: "Low-Level Software Security for Compiler Developers". It also includes a link to the repository's website ([llsoftsec.github.io/llsoftsecbook/](https://llsoftsec.github.io/llsoftsecbook/)), a list of tags (`security`, `compiler`, `book`, `compiler-development`), and a list of contributors (6 total). The repository has 87 stars, 12 watchers, and 8 forks.



# LIVE VERSION AT [LLSOFTSEC.GITHUB.IO/LLSOFTSECBOOK/](https://llsoftsec.github.io/llsoftsecbook/)

## [1 Introduction](#)

### [1.1 Why an open source book?](#)

## [2 Memory vulnerability based attacks and mitigations](#)

### [2.1 A bit of background on memory vulnerabilities](#)

### [2.2 Exploitation primitives](#)

### [2.3 Stack buffer overflows](#)

### [2.4 Code reuse attacks](#)

### [2.5 Non-control data exploits](#)

### [2.6 Hardware support for protection against memory vulnerabilities](#)

### [2.7 Other issues](#)

### [2.8 JIT compiler vulnerabilities](#)

## [3 Covert channels and side-channels](#)

### [3.1 Cache covert channels](#)

#### [3.1.1 Typical CPU cache architecture](#)

#### [3.1.2 General operation of cache covert channels](#)

### [3.2 Timing covert channels](#)

### [3.3 Resource contention channels](#)

### [3.4 Channels making use of aliasing in](#)

# Low-Level Software Security for Compiler Developers



This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

© 2021 Arm Limited [kristof.beyls@arm.com](mailto:kristof.beyls@arm.com)

Version: 0-75-g9f64a94

## 1 Introduction

Compilers, assemblers and similar tools generate all the binary code that processors execute. It is no surprise then that for security analysis and hardening relevant for binary code, these tools have a major role to play. Often the only practical way to protect all binaries with a particular security hardening method is to let the compiler adapt its automatic code generation.

With software security becoming even more important in recent years, it is no surprise to

# CURRENT BOOK STRUCTURE

1. Introduction
2. Memory vulnerability based attacks and mitigations
3. Covert channels and side-channels
4. Physical access side-channel attacks
5. Remote access side-channel attacks
6. Supply chain attacks
7. Other security topics relevant for compiler developers

# LOOKING FOR HELP

The project is at its start - maybe 10% of the content is present. We can use a lot of help and contributions:

- Ideas for new content.
- Any other ideas on how to improve the book.
- Review of existing content and newly written content in pull requests.
- Contributions of new sections or improvements to the text.
- Introduce your friends to this project.

How to reach out:

- Raise a github issue or start a discussion thread
- Reach out to me in any way, e.g. by email at [kristof.beyls@gmail.com](mailto:kristof.beyls@gmail.com)